DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO DATABASE SYSTEMS

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984

Database Systems

The systems in this list were chosen because of their ubiquity, their historical interest, their potential for experimentation, or their significance for further study. Other sources for references to database and file systems can be obtained from commercial software catalogs (DATAPRO, Auerbach, ICP Quarterly) or from surveys in computer magazines, for example, Krass[81]. CODASYL[71A] contains a detailed review of GIS, MARKIV, NIPS/FFS, TDMS, UL/1, COBOL, DBIG, IDS, IMS, and SC-1; Martin in Nance[75] surveyed STAIRS, DIALOG, DATA CENTRAL, ORBIT used for MEDLINE, DARIS, SPIRES, LEADER, RECON, RIQS, INTREX, and NASIS. Kim[79] surveyed relational DBMS' and Brodie[82] includes a survey of relational systems: IDM, INGRES, MRDS, MRS, NOMAD, ORACLE, PASCAL/R, PRTV, RAPPORT, SYSTEM R, QBE, RAPID, and cites a total of 60. Wiederhold[83] includes descriptions of distributed DBMS efforts. CODASYL[78] provides guidelines for system selection. Landau[79] produces a listing of on-line databases.

The commercial systems included below vary in price from several hundred dollars total to several thousand dollars per month.

Name	Year	Developer location	Computer	Type and features
ACCENT R	1981	National Information Syst, Cupertino CA	DEC10/20	Com DBMS slc rel stq sch
ADABAS	1971	software ag Darmstadt FRG	IBM360/370 Siemens	Com DBMS hlc stq sch drf eip pri rec Atre[80]
ADEPT	1969	System Dev Corp Santa Monica CA	IBM360-50	Exp DBMS slc nlq(CONVERSE) pri Weissman[69]
ADMINS	1966	MIT&ADMINS Inc Cambridge MA	DEC 11, VAX	DBMS slc sch rel trf McIntosh[68]
ALPHA	1971	IBM Research San Jose CA		Pro DBL slc rel Codd in Codd[71]

Name	Year	Developer location	Computer	Type and features
AMBASE	1979	Amcor Computer Corp Louisville KY	DEC 11 (RSTS)	Com DBMS sch isf
AMIGOS	1970	Comress Inc Rockville MD	IBM360/370	Com FMS hlc isf
APPEL IV	1974	SIS Paris France	IBM360/370 Burr.500/700	Com DBMS hlc hie drf isf
ASI/INQ	1975	Applications Softw. Torrance CA	IBM360/370	Com QUS for DL/1 slc hie stq
ASI/ST	1969	Applications Softw. Torrance CA	IBM360/370 Univac70	Com QUS slc rpg tbq sch *for* sqf, isf, DMS, TOTAL
Ass.PL	1967	General Motors Warren MI	IBM360-67	Inst DBMS hlp(PL/1) rnf gra Dodd[66]
AUTONOTE	1969	Univ of Michigan Ann Arbor MI	IBM360/370 (MTS)	Exp SATDBMS slc txt sch sqf drf Reitman[69]
BASIS	1970	Batelle Mem Labs Columbus OH	CDC6400 DEC 10/20,VAX IBM 370 UNIVAC1100	Inst IRS stq bib rpg Fried in Walker[71]
BEAST	1968	Brookings Inst Washington DC	DEC POP10	Inst SATDBMS slc sch sqf Kidd[68]
BIS	1967	Am Tel &Tel New York NY	IBM360	Inst DBMS hlc hie sch Benner[67]
CAFS	1976	ICL Stevenage UK		Dev DBCMP rel Babb[79]
CASSM	1975	Univ of Florida Gainesville FL		Exp DBCMP Su[79] Hawthorn[82]
CDMS	1969	System Dev Corp Santa Monica CA	IBM360/370	Com DBMS service slc irq isf *see* TDMS
CDMS	1974	Digital Eq Corp Maynard MA	DEC 11	Com DBMS slc hie trf *see* MUMPS
CFS	1980	Carnegie-Mellon U. Pittsburgh PA	DEC LSI-11s	Exp DFMS
CIA	1982	Computer Invest.Adv. Sewickley PA	Apple	Com DBMS i-rel alg hlc(BASIC) rpg sch isf
COGENT	1969	Comp Sciences Corp Los Angeles CA	IBM7090, 370 Univac1100	Com DPG hlp(COBOL) sch hie isf icf
CONVERSE	1967	System Dev Corp Santa Monica CA	IBM360-67 ANFSQ32	Exp QUS net nlq vrf Kellogg[68], in SIGIR[71]

Name	Year	Developer location	Computer	Type and features
COSTAR	1978	Mass Gen Hospital Boston MA	DEC PDP15,11 Tandem	SATBMS slc trf(MUMPS) Barnett[79]
CREATE	1975	Complete Computer Sys, Horsham PA	DataGeneral	Com DPG ixf pri
CREATE/3000	1977	CRI Inc Mountain View CA	HP 3000	Com DBMS hlc rel slq ixf
CZAR	1970	Crown Zellerbach San Francisco CA	IBM360/370	Inst QUS hlc sch isf Palmer[75]
DATA ANALYZER	1971	Program Prod. Nanuet NY	IBM360/370	Com DRS slc rpg sch sqf isf
DATACATALOG	1974	Synergetics Bedford MA	IBM370 OS,DOS UNIVAC	Com DDICT slc rpg sch isf DMS1100 IDMS IMS S2000
DATACOM	1970	Aplied Data Res. Dallas TX	IBM360/370	Com DBMS hlc sch sqf ixf cpr
DATA COMPUTER	1971	Comp Corp of Am Cambridge MA	PDP10 (TENEX)	Dev DBMS hlc hie stq vrf for ARPAnet
DATAMAN	1975	Dataman Ltd Calgary Alberta	IBM360/370	Com FMS slc rpg sch sqf
DATAMANAGER	1976	MSP London UK & Lexington MA	IBM360/370	Com DDICT slc rpg sch ADABAS IMS MARKIV SYSTEM 2000 TOTAL
DATAMASTER	1980	Microsoft Seattle WA	Apple 8080	Com FMS rpg sch sqf
DATASAAB	1974	Saab-Scania AB Linkoping Sweden	SAAB D22/D23	Com DBMS hlp(COBOL) net sch rnf pri Bubenko[75]
dBASE II	1981	Ashton-Tate Culver City CA	Z-80 CP/M	Com FMS, Join stq rpg isf(1 updated)
DBC	1978	Ohio State&UNIVAC Columbus OH		Dev DBCMP sch isf Banerjee[79], Hawthorn[82]
DBMS	1977	Prime Computer Inc Wellesley Hills MA	Prime	Com DBMS hlp net sch stq(SQL) rnf rec pri
DBMS10/20	1973	Digital Eq Corp Marlboro MA	DEC 10/20	Com DBMS hlc net(1973) sch stq(SQL) rnf rec pri
DBMS11	1979	Digital Eq Corp Marlboro MA	DEC 11	Com DBMS hlc net(1973) sch rnf
DBMS990	1980	Texas Instruments Austin TX	TI990	Com FMS hlc(PASCAL COBOL FORTRAN) hie-tbq isf

Name	Year	Developer location	Computer	Type and features
DBMS1900	1974	ICL London England	ICL 1903	Com DBMS hlc sch isf pri rec
DBOMP	1970	IBM White Plains NY	IBM360/370	Com BOMP net stq DL/1 files; CFMS *for* isf
DBS90	1972	Sperry Rand GmbH Frankfurt a/M FRG	Univac90	Com BOMP hlc(COBOL) net
DB/DC	1975	IBM White Plains NY	IBM360/370 IMS	Com DDICT slc sch rpg (CMIS *for manufacturing*)
DIALOG	1967	Lockheed Res Corp Palo Alto CA	IBM360	Com IRS *service* slc irq hlc bib Walker[71]
DIRECT	1977	Northwestern Univ Evanston IL		Pro DBCMP DeWitt[79] Hawthorn[82]
DISAM	1975	Four Phase Systems Cupertino CA	4phase70	Com FS *for* DDBMS hlp(COBOL) isf ixf
DL/1	1968	IBM White Plains NY	IBM360/370	Com FMS hlc sch hie sqf isf drf stq(CICS)
DM1, 5 *also* SC-1	1966	Auerbach Philadelphia PA	Univac418 IBM360/370	Com DBMS slc hie rpg sch ixf CODASYL[71A]
DMS1100, 90	1971	Univac Minneapolis MN	Univac1100 Univac90	Com DBMS hlp(COBOL) net(1969) rnf rec vie
DMS170	1977	Control Data Corp Minneapolis MN	Cyber170	Com DBMS hlc sch sqf *derived from* (MARS) isf drf
DMS/1700	1975	Dedicated Systems Chicago IL	Burr.1700	Com FMS ixf
DMS II	1972	Burroughs Pasadena CA	B1700 *to* 7700	Com DBMS hlp(COBOL, ALGOL) net sch sql isf ixf cpr rec
DMS IV *based on* IDS	1972	Honeywell Inf Sys Phoenix AZ	H60	Com DBMS hlp(COBOL) net(1973) rpg rnf rec
DPL *also* IPL	1976	National Information Syst, Cupertino CA	DEC10/20	Com DBMS hlc(FORTRAN COBOL) sch isf pri
DYL250	1971	Dylakor Comp Syst Encino CA	IBM360/370	Com FMS slc rpg stq sql isf
EDEN	1982	Univ of Washington Seattle WA	DEC VAX	Exp DFMS obj Jessop in Wiederhold[83]
EDMS+	1969	Control Data Corp Brussels Belgium	CDC6400 Cyber	Com DBMS hlc(FORTRAN) stq sch(ANSII) isf Nijssen[77]

Name	Year	Developer location	Computer	Type and features
FACETS *was* PRISM	1981	Synergistics Bedford MA	IBM 370	Com DDICT slc rpg
FOCUS		Information Builders New York 10001 NY	IBM 370 CMS,TSO	Com QUS irq rpg isf,DL/1,IDMS prl
FORDATA	1974	CSIRO Canberra Australia	CDC CYBER76	Inst DBMS hlp(FORTRAN) net sch
FORIMS	1970	Nippon Univac Tokyo Japan	Univac1100	Dev DBMS hlc(FORTRAN) net ixf rec
FORTE	1959	Burroughs Corp Paoli PA	B2500/3500 B1700-7700	Com FMS hlc(COBOL) sqf isf drf ixf rnf Chapin[69]
FRAMIS	1977	Lawrence Liv.Lab. Livermore CA	CDC 7600 CRAY DEC VAX(VMS)	Dev DBMS rel alg stq rnf(CODASYL)
GIM	1967	TRW Systems Redondo Beach CA	IBM7094,360/ 370 Univac1100 Honeywell6000	Com QUS slc stq sch rec drf rng Nelson[67]
GIS	1966	IBM White Plains NY	IBM360/370	Com QUS hlc(COBOL, PL/1) hie stq sch sqf isf
GMIS	1975	MIT Sloan School& IBM, Cambridge MA	IBM370 (XRM)	Dev SATDBMS(Dec.supp.) rel vrf Donovan[78]
IDBP 80/440	1982	Intel-MRI Austin TX	*links to* IEEE 488, ETHERNET	Com DBCMP rel prl rec
HOPS	1975	Technicon Haifa Israel	Burroughs 126	Exp DBMS hie trf Reiter In Kerr[75]
IDM 500	1981	Britton-Lee Los Gatos CA	VAX *on* IEEE-488 *or* intlgnt.term.	Com DBCMP rel isf,ixf prl
IDMS	1972	Cullinane Corp Westwood MA	IBM360/370 ICL1902 Univac70,90 Siemens4004	Com DBMS hlp(COBOL) nlq(ROBOT) rpg (CULPRIT) net(1973+) sch DDICT drf rec
IDS I, II	1962	Honeywell Inf Sys Phoenix AZ	H200, H60, H6000	Com DBMS hlp(COBOL) net(73) rnf rec Bachman[68]
IDP EDMS	1978	Honeywell was XDS Los Angeles 45 CA	H66 Sigma 6,7,9	Com DBMS hlc(COBOL) net rnf ixf sch rec
IFIP *also* RIM	1978	Boeing Computer Co (IPAD) Seattle WA	DEC VAX IBM	SATDBMS *for* CAD/CAM net(1978) hlp(FORTRAN)
IMAGE	1974	Hewlett-Packard Santa Clara CA	HP3000, HP2100	Com DBMS hlc sch prl stq net(2 level: drf, rnf)

Name	Year	Developer location	Computer	Type and features
IMARS	1971	Computeria Inc Braintree MA	DEC PDP10	Com QUS slc stq rpg rec
IMS-2/VS	1968	IBM White Plains NY	IBM360/370	Com DBMS hlc *multi*-hie sch(DL/1) stq rec(-VS)
INFOS	1975	Data General Southboro MA	DG Nova, Eclipse	Com FMS hlc hie isf ixf stq
INGRES	1973	Un.of CA&Relational Technology, Berkeley	DEC 11, VAX (UNIX, VMS)	Dev DBMS slc hlp(C, ea.) rel stq gra pri Held[75]
INQUIRE	1969	Infodata Systems Falls Church VA	IBM360/370	Com IRS hlc rpg stq sch pri; Dev DDBMS(IQNET)
INTREX	1966	Mass Inst. Tech Cambridge MA	IBM7094, IBM360	Inst IRS irq/stq bib ixf Walker[71]
IS/1 *later* PRTV	1971	IBM UK Research Peterlee UK	IBM360/370	Dev DBMS hlp(PL/1) rel alg vie com stq Todd[78]
ISAM70	1974	Software70 Anaheim CA	Any FORTRAN system	Com FS hlc(FORTRAN) isf
LADDER	1977	SRI International Menlo Park CA	DEC PDP10	Exp IRS nlq net(DDMS20) Hendrix[78]
LEADER	1967	Lehigh Univ Bethlehem PA	CDC6400	Inst IRS irq bib ixf Hillman[69]
LEXICON	1976	Arthur Anderson Chicago IL	IBM360/370, System 3	Com DDICT slc IDMS IMS TOTAL
LEXIS	1978	Mead Data Central Lexis New York NY	IBM370	Com IRS *service* slc *legal, economic databases*
LUNAR	1972	Bolt Beranek Newman Cambridge MA	DEC PDP10 TENEX	Inst IRS nlq sqf ixf Woods[73]
MADAM	1970	MIT MacAIMS Proj. Cambridge MA	H6000 (MULTICS)	Dev DBMS *first* rel hlp (PL/1) stq rpg vrf Strnad[71]
MAGNUM	1975	Tymshare Cupertino CA	DEC PDP10	Com DBMS slc rel stq sch
MARKIV	1967	Informatics Canoga Park CA	IBM360/370 Univac900	Com FMS slc rpg tbq hie isf CODASYL[71A]
MARS	1969	Control Data Corp Sunnyvale CA	CDC6400	Com DBMS hie sch hlc(FORTRAN) sqf isf
MDBS	1980	Micro Data Base Syst Lafayette IN	Z80,8080-*based systems on CP/M*	Com DBMS slc stq net sch

Name	Year	Developer · location	Computer	Type and features
MEDLARS *also* ELHILL	1963	National Lib Med Bethesda MD	IBM360	Inst SADBMS slc irq sqf isf bib Katter[72], *also* ORBIT
MICRO-SEED	1980	Microsoft Bellevue WA	Z80,8080-*based systems*	Com DBMShlc(FORTRAN) net sch *based on* SEED
MODEL204	1972	Comp Corp of Am Cambridge MA	IBM360/370	Com DBMS hlc stq sch plf ixf(IFAM)
MORIS	1972	Polytechnico Milano Italy		Pro DBMS rel cal sch stq(COLARD) Braechi in Klimbie[75]
MUMPS	1966	Mass Gen Hospital Boston MA	DEC PDP15,11 8080 ao.	Com FMS slc hlc trf Greenes[69]
MRDS *based on* RDMS	1978	Honeywell Inf Sys Minneapolis MN	H6000 L68 (MULTICS)	Com DBMS hlp(PL/1) rel stq(LINUS) rpg vrf
NOMAD	1975	National CSS Norwalk CN	IBM370-CMS	Com DBMS slc,hlc hlc stq sch rpg
NYTIS	1970	New York Times New York NY	IBM360	Inst IRS slc txf ixf Baker[72]
OASIS	1971	Stanford Adm DP Stanford CA	IBM360/370	Inst SATDBMS hlc(COBOL) hlc irq isf
ORACLE	1979	Relational Software Inc., Menlo Park CA	DEC VAX, IBM VM MVS DOS	DBMS hlc(COBOL PL/1 C) rel stq ixf cpr pri vlc
OSIRIS	1973	Survey Res.Ctr.,Univ. of Michigan, Ann Arbor	IBM360/370	Inst SATDBMS sch sqf,isf rpg Rattenbury[74]
PLUS/4	1979	Century Analysis Pacheco CA	NCR 101 etc.	Com FMS hlc
POLYPHEME (SIRIUS)	1977	I. N. Polytechnique Grenoble France	CII & IBM	Exp DDBMS pri LeBihan[80]
RAMIS	1967	Mathematica Princeton NJ	IBM360/370	Com DBMS hlc stq rpg
RAP	1975	Univ of Toronto Toronto Canada		Exp DBCMP rel Ozkarahan[77] Hawthorn[82]
RAPPORT	1978	Brit.Min.Def. & LOGICA,New York NY	*Any* FORTRAN-*based system*	Com DBMS hlc stq rpg rec
RDF	1967	Rand Corp. Santa Monica CA	IBM360	Exp IRS slc nlq rel Levien[67]
RDMS *based on* MADAM	1971	Mass Inst of Tech Cambridge MA	H6000 (MULTICS PL/1)	Inst DBMS hlp rel stq rpg vrf Steuert in [Rustin:74]

Name	Year	Developer location	Computer	Type and features
RECIS *early name:* RDMS	1972	General Motors Warren MI	IBM360-67	Dev DBMS hlp(PL/1) rel stq Joyce[78]
REL	1969	Calif Inst of Tech Pasadena CA	IBM360	Dev IRS nlq rel *extendible* Thompson[69]
RELCRAF	1982	Adv.Rel.Techn.Inc. Menlo Park CA	PRIME OS *or* MPX	Com IRS stq rel cal vie txt gra sch lxf
RETRIEVE	1970	Tymshare Cupertino CA	XDS940	Com DBMS service slc stq 1-sqf; FMS(IML) *for >2*
RFMS	1971	Univ of Texas Austin TX	CDC6400	Dev DBMS hie stq sch Hardgrave[80]
RISS	1974	Forest Hosp.& MIT Des Plaines IL	DEC 11	DBMS 1-rel sch rpg sqf McLeod[75]
ROBOT	1973	Software Sciences Farnborough UK	ICL 1906 Univac9400	Com DBMS slc rpg tnf Palmer[75]
RS/1, *based on* PROPHET	1980	Bolt Beranek Newman Cambridge MA	DEC 11, VAX	SATDBMS slc *with* PL/1 tbq rpg grf sch
SAM *basis for* RM, XRM	1968	IBM Scientific Center Cambridge MA	IBM360*modified* 370	ExpFMSrelvrfSymonds[68] *later* RSS(SYSTEM R)
SAS	1972	Univ. N. Carolina & SAS Inst., Raleigh NC	IBM360/370	SATDBMS(*statistics*) slc stq rpg sqf
SIA uses QBE	1975	IBM Research Yorktown Heights NY		Dev DBMS rel tbq(*by example*) Ztoof[77]
SCORE	1969	Programming Methods, New York NY	Any COBOL system	Com FMS hlp(COBOL) hie tbq sqf isf
SDD-1	1978	Comp Corp of Am Cambridge, MA	DEC 10/20	Exp DDBMS slc rel sch DAPLEX Bernstein[81]
SEED	1978	Internat'l Data Base Syst., Philadelphia PA	DEC 11,10/20 IBM370 CDC6000	Com DBMS hlp(COBOL) hlc(FORTRAN) sch rpg net(1973) rnf
SEQUITUR	1981	Pacific Software Berkeley CA	DEC-11,VAX Z-8000 systems	Com DBMS slc,hlc(C) rel tbq lxf rpg txt
SESAM	1973	Siemens München FRG	S4004	Com FMS stq rpg sqf *isf*
SHOEBOX	1970	MITRE Corp Bedford MA	IBM360	Exp SATDBMS slc txt plf stq Glantz[70]
SIDAS	1974	Shipping Res Svc Oslo Norway& Houston TX	IBM360/370 Univac1100 DEC PDP10	Com DBMS net hlp(COBOL) hlc sch Palmer[75]

Name	Year	Developer location	Computer	Type and features
SIR	1977	Scientific Information Retrieval, Evanston IL	IBM360/370 CDC 6000,Cyber	Com IRS stq stat.interface sch net pri
SOCRATE	1970	Univ of Grenoble, CII Louveciennes, France	IBM360-370 CII Iris45,80	Dev, Com DBMS slc sch net stq *vtf*
SOLID	1967	Penn State Univ University Park PA	IBM360-40 -67	Exp DBMS slc cpr rec DeMaine[71]
SOURCE	1979	Telecomputing Co McLean VA	*multiple*	Com IRS *service* slc *business, consumer info, ads*
SPIRES	1967	Stanford Univ Stanford CA	IBM360-67, 370-168	Inst IRS, DBMS slc hie irq bib *trf* ixf pri Schroeder in Kerr[75]
SQL/DS	1981	IBM San Jose CA	IBM370 (DOS)	DBMS hlp(PL/1) rel stq ixf(VSAM) pri vie
STAIRS	1972	IBM Stuttgart FRG	IBM360/370 *with* CICS	Com IRS slc stq txt sqf *drf* isf pri
SWALLOW	1980	Mass. Inst. Tech. Cambridge MA		Exp DFMS obj
SYSIF	1970	CAP-SOGETI Paris 15 France	IBM360/370, CII Iris, Univac1100	Com IRS slc stq sqf isf
SYSTEM 1022	1978	Software House Cambridge MA	DEC 10/20	Com DBMS slc,hlc ixf *timeshared services*
SYSTEM 2000 *or* S2000, S2K	1970	Intel-MRI Austin TX	IBM360/370 Univac1100 CDC6000	Com DBMS hlp(COBOL PL/1) hie stq isf Kroenke[78]
SYSTEM C	1981	Software Clearing House, Cleveland OH	NCR, Criterion	Com DBMS hie net(1978) sch stq *ruf* rec pri
SYSTEM-R	1975	IBM Research San Jose CA	IBM370	Exp DBMS hlp(PL/1) rel stq (SEQUEL) vie Astrahan[76]
SYSTEM-R*	1981	IBM Research San Jose CA	IBM370 VMS	Exp DDBMS hlp rel sch stq vie
TDMS	1968	System Dev Corp Santa Monica CA	IBM360-50 (Adept)	Dev DBMS slc hie irq sch Bleier[68], CODASYL[71A]
TOD	1973	Stanford Univ. Stanford CA & ITTRI Chicago 16 IL	IBM360-50,67, 370 DEC VAX	Inst SATDBMS hlc(PL/1) sch irq *gra* ixf tnf epr cip Wiederhold[75]
TOOL-IR	1974	Univ of Tokyo Tokyo Japan	HITAC8800	Inst IRS slc stq bib pri Yamamoto[75]

Name	Year	Developer location	Computer	Type and features
TOTAL	1971	Cincom Inc Cincinnati OH	IBM360/370 Univac 90,V70 CDC Cyber	Com DBMS hlc sch net (2 level:drf,rnf) DDICT Cagan[73]
		also on Siemens4004 Honeywell200 NCR Century		
TRAMP	1967	Univ of Michigan Ann Arbor MI	IBM360-67	Exp DBMS nlq rel Ash[68]
UCC TEN	1976	University Comp. Dallas TX	IBM360/370 IMS	Com DDICT slc rpg sch IMS
UNIDATA	1970	United Computing Kansas City MO	CDC6400	Com DBMS hlc sch rpg gra
VISIFILE	1982	Visicorp San Jose CA	IBM PC, Apple	Com FMS slc ixf stq
WOODSTOCK	1979	Xerox Research Lab Palo Alto CA	Xerox Altos	Dev DFMS Swinehart[79]
ZETA *also* TORUS	1974	Univ of Toronto Toronto Ontario	IBM360/370	Exp DBMS hlc(PL/1) rel stq vie drf Tsichritzis[77]

Legend for type and features of database systems

alg relational algebra
bib bibliographic data
BOMP bill-of-materials processor
cal relational calculus
cip ciphering
Com commercial
cpr compression
DBCMP database computer
DDL database language
DBMS database-management system
DDBMS distributed DBMS
DDICT data dictionary system
Dev developmental
DFMS distributed FMS
DPG database program generator
drf direct or immediate file organization
Exp experimental
FMS file-management system
FS file system
gra graphic data support
hie hierarchical database organization
hlc host-language system accessed by CALL
hlp host-language system with preprocessor
Inst institutional
irq interrogative query processor
IRS information-retrieval system
isf indexed-sequential file
ixf indexed files
net network database organization, (year indicates CODASYL standard)
nlq natural language query capabilty
obj object based
plf pile file organization
pri privacy protection
Pro proposed
QUS query and update system
rec recovery support
rel relational database organization
rnf ring or chain file organization
rpg report generator
SADBMS single-application DBMS
SATDBMS single-application type DBMS
sch schema
slc self-contained system
sqf sequential file organization
stq statement-oriented query processor
tbq tabular query processor
tnf transposed files
trf tree-structured files
txt textual data
vie support for multiple user views
vrf virtual file support

DISEÑO DE ESTRUCTURAS DE DATOS

A R T I C U L O

BEHAVIORAL AND ORGANIZATIONAL CONSIDERATIONS IN THE DESIGN OF INFORMATION SYSTEMS AND PROCESSES FOR PLANNING AND DECISION SUPPORT

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984

Behavioral and Organizational Considerations in the Design of Information Systems and Processes for Planning and Decision Support

ANDREW P. SAGE, FELLOW, IEEE

Abstract—Determinants of performance of systems and processes for planning and decision support are discussed. This paper is directed at people who design such systems and processes, who use such systems and processes, and who manage organizations in which these may be used. The literature cited is associated with several areas including psychology, organizational behavior and design, information science, management science, computer science, and related disciplines. Performance determinants and design requirements for systems and processes for planning and decision support are especially stressed. A number of areas where additional research appears needed are mentioned, and some recommendations and interpretations are given concerning both contemporary efforts and needed future efforts.

I. Introduction

THAT there is much interest in planning and decisionmaking efforts to determine effective public and private sector policies is made evident by the number of recent texts and case studies devoted to these topics [2], [4], [13], [18], [20], [21], [44], [45], [48], [51], [80], [84]–[86], [89], [104], [105], [108], [134], [135], [139], [141], [150], [178], [179], [198], [212], [219]–[222], [237], [243], [283], [293], [318]–[320], [334], [359], [361], [363], [377], [394], [397], [398], [400], [412]. These in part, concern the numerous complexities associated with practical implementation of the results of systemic efforts for planning and decision support. Advances in digital computer technology coupled with advances in systems science, systems methodology and design, and systems management suggest extension of the information analysis and display capability provided by management information systems to include interpretation and aggregation of information and values such as to result in decision support systems (DSS) or planning and decision support systems. There is a growing literature in this area [5], [36], [39]–[41], [76], [86], [110], [133], [138], [224], [226], [227], [239], [240], [258], [309], [350], [356], [366], and this indicates much contemporary interest and activity.

There are a number of requirements for design success with respect to systems for planning and decision support. These involve a considerable number of disciplines. The result of not making appropriate use of pertinent contributions from a number of disciplines in the design of systems for planning and decision support is likely to be a system or process that is deficient in one or more important ways. The purpose of this effort is to discuss, from a systems engineering perspective, some of the many requirements for design success in this area.

It is possible to disaggregate planning and decisionmaking processes into a number of steps. In essence, they are purposeful futuristic efforts which involve the entire systems engineering process [301]–[305], [307], [308] and can, therefore, be described by any of a number of frameworks for systems engineering such as the three-or the seven-step framework which involves:

1) Formulation of the issue.
   a) problem definition (determination of needs, constraints, alterables)
   b) value system design (determination of objectives and objectives measures)
   c) system synthesis (identification of possible decisions or action alternatives and measures of the accomplishment of these);
2) Analysis of the issue.
   d) systems analysis and modeling (determination of the structure of the decision situation, the impacts of identified decisions or action alternatives and the sensitivity of these to possible change in conditions)
   e) optimization or refinement of alternatives (adjustment of parameters or activities such that each identified decision is the best possible in accordance with the value system);
3) Interpretation of the issue
   f) evaluation and decisionmaking (each possible decision alternative is evaluated, prioritized, and one or more alternatives are selected for implementation action)
   g) planning for action (commitment of resources are made and implementation is accomplished).

Janis and Mann [177] have identified a four-stage model of the decisionmaking process. Fig. 1 presents a slightly modified version of this decision process model. We note that it contains the same essential steps involved in the systems engineering process. Of particular interest are the questions asked at each step of the process. We will elaborate upon this model and other models of the decisionmaking process in our efforts to follow.

Comprehensive efforts involving decisionmaking will be complex because of the many disciplines and areas involved as well as because of the subject matter itself. Probably the formal study of decisionmaking first began with the rational economic man concepts of the 18th century mathematicians Cramer and Bernoulli who explained the St. Petersburg paradox. Since then there have been many workers from a large number of disciplines who have been concerned with various types of decisionmaking studies and the provision of assistance to enhance the understanding of rationale for plans and decisions as well as improvements in the efficiency, effectiveness, and equity of the resource allocations that constitute planning and decisionmaking.

Contemporary choicemaking issues in the public and private sector are complex, contain much uncertainty, and require inputs from many sectors for full understanding and resolution. Many writers have indicated bounded rationality limits in decisionmaking that would appear to make provision of information system

Manuscript received April 1, 1981; revised June 21, 1981. This work was supported in part by the U.S. Office of Naval Research under Contract N0014-80-C-0542.

A. P. Sage is with the Department of Engineering Science and Systems, University of Virginia, Charlottesville, VA 22901.

0018-9472/81/0900-0640$00.75 ©1981 IEEE

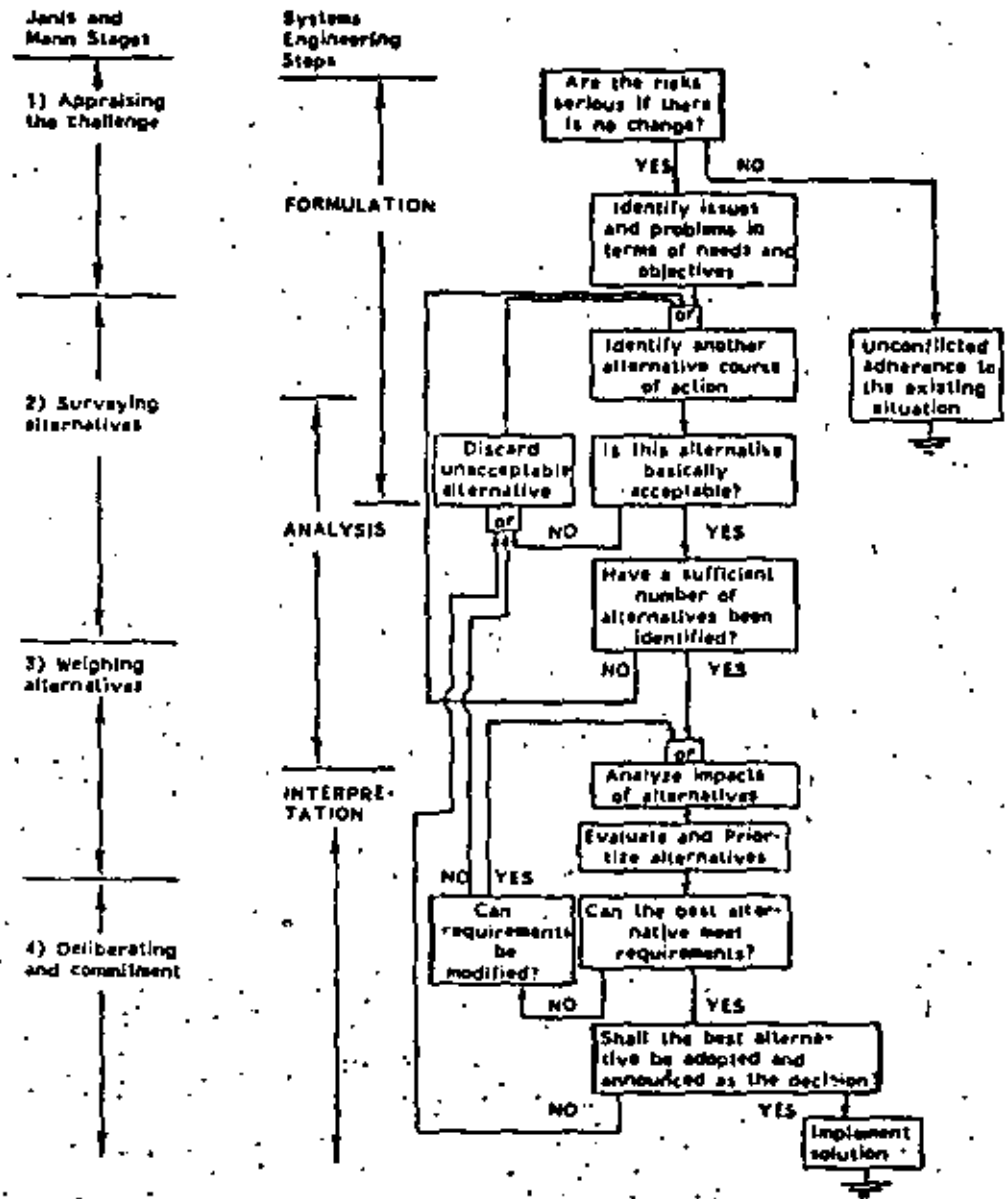


Fig. 1. Systems engineering interpretation of the decision process model of Janis and Mann.

adjuvants for choicemaking normatively very desirable. Such planning and decision support systems could, in principle, provide decisionmakers with rapid access to the information and knowledge needed to enhance decision quality. Unfortunately this promise of enhanced decision quality has not always been realized in practice. There are, doubtlessly, a number of causitive factors inhibiting the potential benefits possible from information systems for planning and decision support. Principal among these factors which, at present, pose fundamental limits to information system success appear to be

1) the need to insure substantive or input–output rationality, such that evaluations of plans and decisions are veridical;
2) the need to insure process rationality, such that the information system accommodates the capabilities of, and the constraints placed upon, the user;
3) the need to understand and cope with human cognitive limitations as they affect the formulation, analysis, and interpretation of decision situations and alternatives; and
4) the need to understand and integrate the normative or prescriptive components with the descriptive components of decision situations in order to evolve realistic adjuvants for the formulation, analysis, and interpretation of decision options.

This paper presents a survey, status report, and integration and interpretation of research from a diversity of areas that supports the design of information systems capable of coping with the needs and fundamental limits to improved judgment. We discuss and describe

1) the cognitive styles of decisionmakers;
2) individual human information processing in decision situations and biases in the acquisition, analysis, and interpretation of information;
3) decision rules for individual decision situations;
4) contingency task structural models of decision situations; and
5) decisionmaking frameworks, organizational settings, and information processing in group and organizational decision situations.

In a very real sense the structural models section, Section V, is the principal portion of this effort. It contains the basic decisionmaking paradigm, including action selection. The contingency task structure, which comprises the issue at hand, the environment into which the issue is imbedded, the decisionmaker, and decisionmaker experiential familiarity with the issue environment, is the determinant of cognitive style perfor-

mance objectives for the particular task at hand. These, in turn, influence selection of an information processing approach and selection of a decision rule.

The literature in this area is enormous. But there is the need for efforts to integrate it from the perspective of systems engineering design of information systems for planning and decision support. There are a number of recent surveys available that discuss one or a limited number of the topics important for the design of planning and decision support systems. These include the surveys of Barron [27], Benbasat and Taylor [35], Bettman [37], Craik [67], Dunnette [87], Einhorn, Kleinmuntz, and Kleinmuntz [95]; Einhorn and Hogarth [98], Ericsson and Simon [99], Hammond, McClelland, and Mumpower [142]; Hammond [143], Hogarth [159], Hogarth and Makridakis [161], Johnson and Huber [179], Kassin [190], Keen [193], Libby and Fishburn [214], Libby and Lewis [215], Mintzberg [248], Nisbett and Ross [264], Nutt [266], Robey and Taggart [292], Sage and White [304], Schneider and Shiffrin [313], Slovic and Lichtenstein [345], Slovic, Fishhoff, and Lichtenstein [348]; Svenson [365], and Zmud [415]. This work attempts a selective integration of this voluminous literature and extensions and interpretations of it from the perspective of ultimate potential usefulness for the design of information systems for planning and decision support. Generally, references are provided only to published literature of the last half decade with limited references to earlier seminal literature and reports. This was felt desirable in order to limit the reference list to an almost manageable size. Despite our attempt to make this report comprehensive, it doubtlessly fails to incorporate the important contributions of a number of authors. And there are doubtlessly unintentional misattributions and misinterpretations as well. For this apologies are offered and forgiveness requested.

II. Cognitive Styles

It is becoming increasingly clear that it is necessary to incorporate not only problem characteristics, but also problem solver or decisionmaker characteristics, into the design of information systems for planning and decision support. A deficiency in some past designs has been the neglect of the human decisionmakers' role and characteristics and their effects. Essentially all available evidence suggests that problem characteristics and user characteristics influence the planning and choice strategies adopted by the decisionmaker. This section discusses a number of cognitive style models from these perspectives.

Mason and Mitroff [238] have suggested that each person possesses a particular specific psychological cognitive style or "personality" and that each personality type utilizes information in different ways. In their research on (MIS) management information system design, they claim that an information system consists of a person of a certain psychological type who faces a problem in some organizational context for which needed evidence to arrive at a solution is made available through some mode of presentation.

There are five essential variables in the information system characterization of Mason and Mitroff. Each of these are disaggregated into subelements. Mason and Mitroff characterize the *psychological-type variable* according to the Jungian stereotypology. In this typology, people differ according to their preference for information acquisition and analysis and the preferred approach to information evaluation and interpretation. At extremes in the information acquisition dimension are sensing-oriented or sensation types who prefer detailed well-structured problems and who like precise routine tasks, and intuitive-oriented type people, who dislike precise routine structured tasks and perceive issues wholistically. At extremes in the information evaluation dimension are feeling-oriented people, who rely on emotions, situational ethics, and personal values in making decisions; and thinking-oriented individuals, who rely on impersonal logical arguments in reaching decisions.

Mason and Mitroff characterize the *problem variable* into structured and unstructured problems. These may be further divided into decisions under certainty, decisions under risk, and decisions under uncertainty. The *organizational context variable* is characterized as strategic planning, management control, and operational control. The *method-of-evidence-generation variable* involves five types of inquiry systems: the data-based Lockean inquiry system, the model-based Leibnitzian inquiry system, the multiple model-based Kantian inquiry system, the conflicting model-based Hegelian inquiry system, and the learning system based Singerian-Churchmanian inquiry system [254]. A fifth variable, *mode of presentation*, includes personalistic modes of presentation such as one-on-one contact as in drama and art and impersonalistic modes such as abstract analytical models and company reports. These latter four variables do not formally relate to cognitive styles, and some further comment on them is contained in other portions of our effort. A number of works by Mason and Mitroff and their colleagues discuss various aspects of this categorization [252]–[254]. Of interest in this regard is a work by Kilmann [200] which suggests the design of organizations with the Jungian personality characteristics of individuals.

Among the many other studies which have emphasized the need to incorporate decisionmaker characteristics into information system design is that of Doktor and Hamilton [78]. They studied the influence of cognitive style on the acceptance of management science recommendations and found a strong correlation between the decisionmaker's cognitive style and willingness to accept these recommendations. They found that differences in acceptance rates were due not only to differences in cognitive style but also to differences in this subject population. From this and many other investigations [34], [74], [77], [79], [101], [151], [166], [174], [229], [252], [253], [263], [267], [268], [311], [330], [369], it appears that appropriate consideration of the human behavioral variable of cognitive style is very necessary for successful design of decision support systems.

A number of studies such as those by Taylor [369], Craik [67], Payne [272], Schneider and Shiffrin [313], [327] and Simon [342], indicate, as we will discuss in later sections, that human decisionmakers attempt to bring order into their information processing activities when confronted with excess information or the lack of sufficient information. Many early studies assumed that static fixed patterns of dealing with information were "preferred" by the decisionmaker for the process of experiencing the world, and these were referred to as "cognitive style." Some early studies view cognitive style as a mode of functioning that is static and pervasive throughout a person's perceptive and intellectual activities. A number of intellectual processes are subsumed within the term cognitive style. These concern the way in which information is acquired or formulated, analyzed, and interpreted. Thus, cognitive style includes such human activities as information filtering and pattern recognition.

Zmud has indicated [414], [415] that those individual differences which influence information system success most strongly involve cognitive style, personality, and demographic/situational variables. Cognitive style refers to the process behavior that individuals exhibit in the formulation or acquisition, analysis, and interpretation of information or data of presumed value for decisionmaking. Doubtlessly cognitive style is somewhat influenced by such personality variables as dogmatism, introversion, extroversion, and tolerance for ambiguity. However, little appears known concerning these influences. Gough discusses personality and personality assessment in his chapter [87]; but it is rare to find, with some notable exceptions [249]–[251], [330], [332], [352], discussions of personality effects upon decisionmaking behavior in cognition studies. The demographic/situational variables involve personal characteristics such as intellectual ability, education, experience with and knowledge of specific contingency tasks, age, and the like. An important situational variable is the level of stress encountered by the decisionmaker in a

specific problem situation. The level of stress, which results in the adoption of a coping pattern, influences the decisionmaker's ability in acquisition and processing of the information necessary for decisionmaking. The subject of stress will be dealt with in some detail in Section V. Many variables are especially important for an information processing model of cognitive behavior. Some will be discussed in Section III. Our efforts in this section will be devoted primarily, therefore, to cognitive style concepts, especially the role of personality variables in the adoption of cognitive style.

There are a number of cognitive style models in addition to that of Mason and Mitroff. Bariff and Lusk [24], for example, have discussed three cognitive style characteristics relevant to information system design: cognitive complexity, field dependent/independent, and systematic/heuristic. The cognitive complexity characteristic involves three structural characteristics of thinking and perception: differentiation, the number of dimensions sought or extracted and assimilated from data discrimination; the fineness of the articulation process in which stimuli are assigned to the same or different categories; and integration, the number and completeness of interconnections among rules for combining information.

Benbasat and Taylor [35] note that much cognitive complexity research deals with interpersonal perception and has limited value for modeling activities of managers in processing information and making decisions. Mischel is especially perceptive in discussing the potential hazards of attributions and enduring categorizations of people into fixed slots on the basis of a few behavioral signs in his study of the interface between cognition and personality [251]. The assumptions that static characterizations are sufficiently informative to enable behavior predictions in specific settings are strongly challenged. An evaluation of the uses and limitations of static trait characterization of individuals is presented and the strong interacting role of context is emphasized. Mischel is especially concerned with "cognitive economies," that is to say the recognition that people are easily overloaded with an abundance of information and that simplified methods of acquisition and processing of information are, therefore, used. He is especially concerned also with growth of self-knowledge and rules for self-regulation with maturation, topics to be discussed in Section V. We concur with these views in that we believe that it is the individual's experience with the task at hand that is the primary determinant of cognitive style. Further we believe that it is an individual's information processing capacity under various levels of stress and in different contingency task structures that determines, in part, the quality of decisionmaking. These factors depend strongly upon experience. Thus we support the information processing view of Simon [337]–[344] that few characteristics of the human information processing system are invariant over the decisionmaker and the task. These characteristics are generally experiential and evolve over time in a dynamic fashion. They are not static and can not be treated as static and task invariant for a given individual.

In the Bariff and Lusk cognitive style model [24], individuals may be categorized according to whether they are tightly bound by external referrents in structuring cognitions, in which case they are called field dependent or low analytic; or whether they can make use of internal referrents as well as external referrents in structuring cognitions, in which case they are high analytic or field independent. In a field-dependent mode, perception is dominated by the overall organization of the field. There is limited ability to perceive discrete parts of a field, especially as distinct from a specific organized background. Field independent people have more analytical and structuring abilities in comparison to field dependent people in that they can disaggregate a whole into its component parts.

The systematic-heuristic categorization of Bariff and Lusk describes cognitive styles associated with people who either search information for causal relationships that promote algorithmic solutions, or who search information by trial and error hypothesis testing. Systematic individuals utilize abstract logical models and processes in their cognition efforts. Heuristic individuals utilize common sense, past experience, and intuitive "feel." Systematic individuals would be able to cope with well-structured problems without difficulty and would approach unstructured problems by attempting to seek underlying structural relations; whereas heuristic individuals would attempt to cope with unstructured problems without a conscious effort to seek structural identification.

Of particular importance with respect to cognitive styles are relationships between the environmental complexity of the contingency task structure and information processing characteristics. A number of authors have attempted experiments based on the hypothesis that the conceptual structure of the individual determines information processing characteristics. Conceptual structure is typically measured on a dimension of abstract versus concrete. Abstract individuals would be capable of using integratively more complex conceptual processes than concrete type individuals. Abstractness may be characterized by the ability to differentiate a greater variety of information and to discriminate and integrate information in complex ways. Abstract individuals would, therefore, be expected to base actions on more information and to develop more complex strategies for information evaluation than concrete individuals. This is somewhat similar to Piaget's account of evolving cognitive development,[1] in that the "formal" thinker is capable of abstract thought whereas the "concrete" thinker relies more on preceptual experience as a basis for thought and problem solution. While the work of Piaget appears to assume that cognitive capacity evolves over time, some research involving personality and cognitive style assumes that an individual's cognitive style is not task dependent and not subject to change as a function of contingency variables such as experience.

Among other efforts, Driver and Mock [83] developed decision-style theory, a set of four decision styles based upon the heuristic–analytic characterization of Huysman [172] to relate conceptual structure of decisionmakers to both the amount of information they tend to use and the degree of focusing that they exhibit in the use of information. A heuristic person will use intuition, past experience, concrete thought, and a wholistic approach to reach decisions. An analytic person will utilize abstract logical models and will search for causal relationships and underlying structure to evolve rationale for decisionmaking. The four decision styles are determined by the degree of focus in the use of information and the amount of information desired. A decisive person is one who wishes to see the minimum possible amount of information and who will likely identify a single workable decision. Decision speed obtained from short summary, often verbal reports, is a characteristic of the decisive person. A flexible person is one who utilizes minimum information but who will identify a number of potentially acceptable decisions. A hierarchic person is one who utilizes much information, often obtained in a thorough way from long involved precise reports to identify a single acceptable decision. An integrative person utilizes much information to identify a number of potentially acceptable decisions.

Vasarhelyi [389] has also examined the analytic–heuristic dimension. His experimental results indicate generally that analytic type people tend to use computers and other analytic tools more in planning than do heuristic types. Heuristic types use less information than the analytic types and are more concerned with the lack of flexibility in computers than analytic types. However, his study of correlations among various style-measuring instruments indicates that these are relatively low.

Driver and Mock also suggested a fifth style which they referred to as the complex style, which is characterized by a wide search and analysis of information. It is a mixture of the integra-

[1] See Section V of this paper.

TABLE I
FOUR MODELS OF COGNITIVE STYLE

BARIFF AND LUSK [24]	DRIVER AND MOCK [81]
COGNITIVE COMPLEXITY	DEGREE OF FOCUS IN USE OF INFORMATION
· DIFFERENTIATION · DISCRIMINATION · INTEGRATION	· MULTIPLE SOLUTIONS IDENTIFIED · ONE SOLUTION IDENTIFIED
FIELD INDEPENDENT/DEPENDENT SYSTEMATIC/HEURISTIC	AMOUNT OF INFORMATION USED
	· MAXIMUM · MINIMUM
MCKENNEY AND KEEN [242]	MASON AND MITROFF [238]
INFORMATION ACQUISITION	INFORMATION ACQUISITION
- RECEPTIVE - PRECEPTIVE	- INTUITIVE - SENSING
INFORMATION EVALUATION AND INTERPRETATION	INFORMATION EVALUATION AND INTERPRETATION
- SYSTEMATIC - INTUITIVE	- THINKING - FEELING

tive and hierarchic types. Zmud [415] has performed some experimental studies of this decision style theory. His findings indicate that perceptual differences can indeed be observed for specific cognitive styles and among subjects with different educational and experiential backgrounds. However, his results also indicate that there is no apparent relationship between cognitive style perceptions and actual cognitive behavior despite consistent differences in perceptions of cognitive styles.

McKenney and Keen [242] have done extensive work on cognitive style measurements. These have become, in part, the basis for several definitive efforts [192], [193], [258] in decision support system design. They conceptualize cognitive style in two dimensions: information acquisition and information processing and evaluation. The information acquisition mode consists of receptive and preceptive behavior, both at the opposite extremes of a continuum. They claim that preceptive decisionmakers use concepts, or precepts, to filter data, to focus on patterns of information, and to look for deviations from or conformities with their expectations. Receptive people tend to focus on detail rather than patterns and derive implications from data by direct observation of it, rather than by fitting it to their own precepts.

With respect to information processing and evaluation, McKenney and Keen measured individuals on a scale, with the systematic thinker at one extreme and the intuitive thinker at the other extreme. They have shown, using a battery of pencil and paper tests, that systematic thinkers approach a problem by structuring it in terms of some method which would lead to a solution, whereas intuitive thinkers use trial and error, intuition, and previous experience to obtain solutions.

We have examined four cognitive style characterizations in this section. Table I summarizes the models of cognitive style that result from these efforts. We note the considerable similarity among these four constructs. There have been a number of studies of the measuring instruments involved in classifying people according to these cognitive styles. Many, such as the study by Vasarhelyi [389] mentioned previously, have found rather low correlations among test instruments. Zmud [413], [415] has indicated low correlation also among test scores on different instruments indicating cognitive styles. Chervany, Senn, and Dickson [57], [76] have expressed much concern and pessimism concerning the validity of much of the contemporary research in this area. They comment that the study of individual personality differences as predictors of human behavior and performance have been basically unsuccessful in that it has not been possible to predict performance on the basis of personality characteristics. Their comment and the comment of others that the characteristics of the task in which the individual involved is a prime determinant of human behavior, appears unassailable. We will provide and discuss additional evidence supporting a dynamic cognitive style characterization that will incorporate the contingency task structure and the decisionmaker's task experience in several other sections of this paper. In particular we emphasize the strong need for consideration of the structure and the content of planning and decision situations in order to evolve contextually meaningful support.

III. INFORMATION PROCESSING

Problem solving, judgment, and decisionmaking imply both thought and action. Hence decisionmaking can be defined as the processes of thought and action involving an irrevocable allocation of resources that culminates in choice behavior. In making a decision, more often than not, the decisionmaker is dealing with environments characterized by risks, hazards, uncertainty, complexity, changes over time, and conflict. Further, the quality of a decision depends upon how well the decisionmaker is able to acquire information, to analyze information, and to evaluate and interpret information such as to discriminate between relevant and irrelevant bits of data. Decision quality also depends upon how well the decisionmaker is able to cope with stress, which is invariably encountered in important decision circumstances. Effective management of these factors enables strategies by which the decisionmaker may arrive at a good problem solution, decision, or judgment.

A number of studies such as those by Barron [28], Bettman [37], Chorba and New [58], Delaney and Wallsten [73], Feather [107], Howell and Fleishman [165], O. Huber, [168], G. Huber [170], [171]; Ives, Hamilton, and Davis [174]; Libby and Lewis [215], Lucas and Nielson [228], MacCrimmon and Taylor [232], Montgomery and Svenson [256], Moskowitz, Schaefer, and Borcherding [259], Payne [275], Simon [342], Tushman and Nadler [379], Tuggle and Gerwin [380], Wallsten [395], [396]; and Wright [402]–[406] discuss the vital role of human information processing in decisionmaking. Most contemporary researchers regard information processing as a crucial task for effective decisionmaking and state that the type of decision problem, the nature of the decision environment, and the current state of the decisionmaker combine to determine decision style and decision strategy for a specific task. The term information processing refers to the processing of verbal reports as well as quantitative data since verbal reports are data [99].

An information processing theory of problem solving, judgment, and decisionmaking is based on the assumption that individuals have an input mechanism for acquisition of information, an output mechanism for interpretation and choicemaking, internal processes for filtering and other analysis efforts associated with information, and memories for long- and short-term storage of information. There are a large number of ways of representing

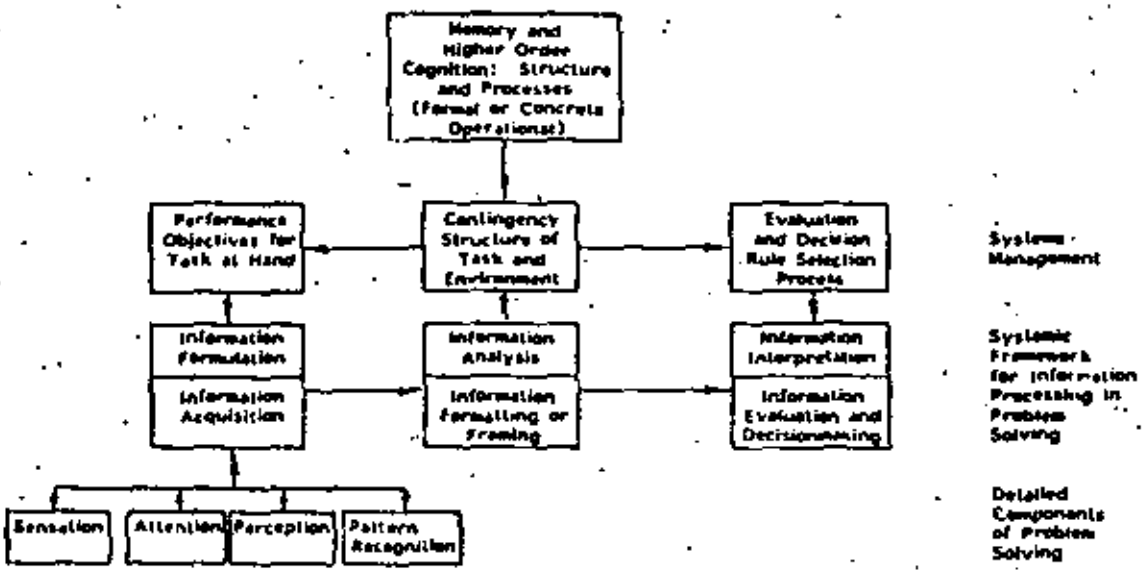


Fig. 2. A systems engineering conceptual model of human information processing.

human information processing. Many of these are described in texts in cognitive psychology such as Anderson [7], Posner [281], or Solso [354], and in works in consumer choice such as Bettman [37]. Much of the work in this area owes a great deal to Simon [334]–[344] who has developed information processing theories in psychology and in artificial intelligence.

Fig. 2 presents a conceptual model of a systems engineering framework [308] for human information processing. There are doubtlessly a number of components missing from this model. It does not show, for example, the essentially iterative nature of the process. Nevertheless we feel that it provides a useful point of departure and a structure for our efforts to follow.

The key functions, which determine how a specific problem or decision situation is cognized, depend upon an interaction of the memory and higher-order cognition of the problem solver with the environment through the contingency task structure. We will be very concerned with development of a conceptual model of higher-order cognition and the contingency task structure in Section V. It is appropriate to remark here that the various information analysis and interpretation processes of thinking, task performance objective identification, evaluation, and decision rule identification, are called "higher order" cognition. This is not because they are somehow more important than the so called "lower order" cognition efforts of information acquisition involving formulation: sensation, attention, perception, and pattern recognition; but because they occur later in time in the overall information processing effort.

It is important to note that information processing and decisionmaking efforts intimately involve memory. Memory [102] influences human judgment in a number of ways. It will influence the perception of the contingency task structure associated with an issue as well as the decision rules used for evaluation of alternatives. Two characteristics of human memory are of special importance for our efforts here. First, information will be encoded in more or less efficient and effective ways in terms of human abilities for recall. The coding process is dependent, also, upon the interpretation attached to information and this strongly influences event recall, perceptions, and associated cognitive biases. The literature concerned with memory and its components, and their relations and interaction with human perceptual experience and behavior is vast and speculative in nature. There have been many studies, both physiological and psychological, concerned with the identification of the memory "engram," which is hypothesized to be the fundamental unit of memory. We need not be especially concerned in this effort with the various physiological structures and processes associated with human memory; or with various related behavior therapies [109]; however the essentials are reviewed below briefly. A useful brief survey of the literature on memory is presented by Thomassen and Kempen in chapter 3, vol. II [244], by Fox [128], and by Radcliff [284].

Human memory constitutes two major components, short-term memory and long-term memory. Short-term memory plays a key role in immediate recall of actively rehearsed limited information [7], [354]. Unless conscious effort is put forth in recalling information from short-term memory, this cannot be done after a lapse of 30 to 60 s from initial presentation. Models of a working short-term memory involve a number of mechanisms, such as an articulatory rehearsal loop that has the capacity to retain short verbal sequences. This is just one mechanism by which short-term retention is possible. There are a number of other sensory registers. It is important to note that short-term memory is an integrated network of many mechanisms, and is associated, in use, with a number of skilled processes.

Shiffrin and Schneider [313], [327] incorporate concepts of attention, memory, and perceptual learning in their theory of short-term retention. They hypothesize short-term storage, the function of which is active control of thinking, reasoning, and general memory processes. According to Shiffrin & Schneider, short-term storage is an activated subset of long-term storage. Transfer of information from short-term storage to long-term storage is dependent on attentional limitations, interference from strong external and internal stimuli, extent of analysis of information, and formation of associations in long-term storage. There have been many studies involving concepts such as retrieval processes, memory trace identification, encoding processes, and recognition which we will not discuss as they appear of secondary importance to the goals of this particular effort. While five to seven unconnected items is believed to be the maximum amount of information that can be retained in short-term memory, long-term memory may contain a virtually limitless amount of information.

Thus we see an enormous difference between human abilities and computer abilities. Because of its large long-term memory and ability for quick search and recall, a human mind easily reasons wholistically. Wholistic reasoning, such as reasoning by analogy, is not at all easy, at this time at least, for a computer. Significant unaided computational effort would be difficult for a human since computation must be done in short-term memory. There exists the possibility that information stored in long-term memory is flawed because of cognitive-biases introduced by processing in short-term memory. A principal task of computer-aided support must be to augment human capabilities in need of

augmentation, while not diminishing abilities in those areas in which human abilities exceed those of the computer [81].

Our effort in the remainder of this section will be devoted to a description of the various processes which support information acquisition and information analysis. We will also discuss some of the cognitive biases that can result from "poor" information acquisition and information analysis. Information interpretation, which leads to alternative evaluation and decisionmaking, is an important and somewhat distinct part of the overall information processing model. It will be discussed in the next four sections from several perspectives.

The types of operations involved in information acquisition are sensation, attention, perception, and pattern recognition. Doubtlessly there are other valid ways of categorizing these operations [7], [37], [67], [100], [137], [148], [175], [281], [297], [298], [313], [327], [333] but the taxonomy used here is sufficient for our purposes. In sensation, information is acquired through the five major sense modalities, which are environmentally activated in response to a specific array of stimulus energies. In a specific decisionmaking situation, the decisionmaker filters out bits of data believed to be irrelevant. The filtering process is based upon task characteristics, experience, motivation, as well as other features and demands of the specific decisionmaking situation. If such a filtering mechanism were not to exist, the decisionmaker would often encounter information overload which generally results in saturation and the inability to process sufficient information for the task at hand. Short-term and long-term memory components play key roles in the information acquisition process as the decisionmaker proceeds with efforts that culminate in choice. A response system couples the memory system to the sensory system and the environment. Thus it controls or activates the sensory modalities on the basis of the actions taken. Through the response system we close the information flow feedback loop. Bower, in volume I of Estes [100], has summarized principal components of the flow system. A model of the principal components of information flow might consist of: the response system, the sensory system, the memory system, and the central processor. The central processor coordinates memorizing, thinking, evaluation of information, and final decisionmaking.

Ultimately involved in retention processes is the notion of attention [7]. In order for information to be transferred from short-term memory to long-term memory, constant conscious attention, in terms of rehearsal, is required. Information entering short-term memory that is not attended to, through specific conscious processes, is lost. Processing of information demands attending to relevant bits of incoming data and transfer of the data into long-term memory for future retrieval for making a decision. Interferences of various types may interrupt attention and thus hinder transfer and retention of relevant stimuli into long-term memory.

Inherent in the processing of information acquisition, is the process of pattern recognition. This process generally involves two phases: extraction and identification. A given stimulus is "coded" in terms of its features. These extracted features of the object or stimulus describe the stimulus. The term "features" implies such characteristics as angles, lines, or edges. A stimulus may be received through any of the sense modalities. The meaning that this conveys to the decisionmaker, or the manner in which the decisionmaker perceives the stimulus, is dependent upon the patterns extracted from the stimulus. In the identification phase, the sensory–perceptual system classifies the stimulus object. The way in which this is often assumed to occur is by a weighted matching of the current feature list against a likely set of prototypes in long-term memory [7], [313], [327], [354] with the input being classified according to the name of the best matching prototype. The quality or extent of the sensory information extracted determines the accuracy of identification.

Thus pattern recognition processes involve memory and the other three components of information acquisition: sensation, or the initial experience of stimulation from the sensory modalities; attention, or the concentration of cognitive effort on sensory stimuli; and perception, or the use of higher-order cognition to interpret sensory stimuli.

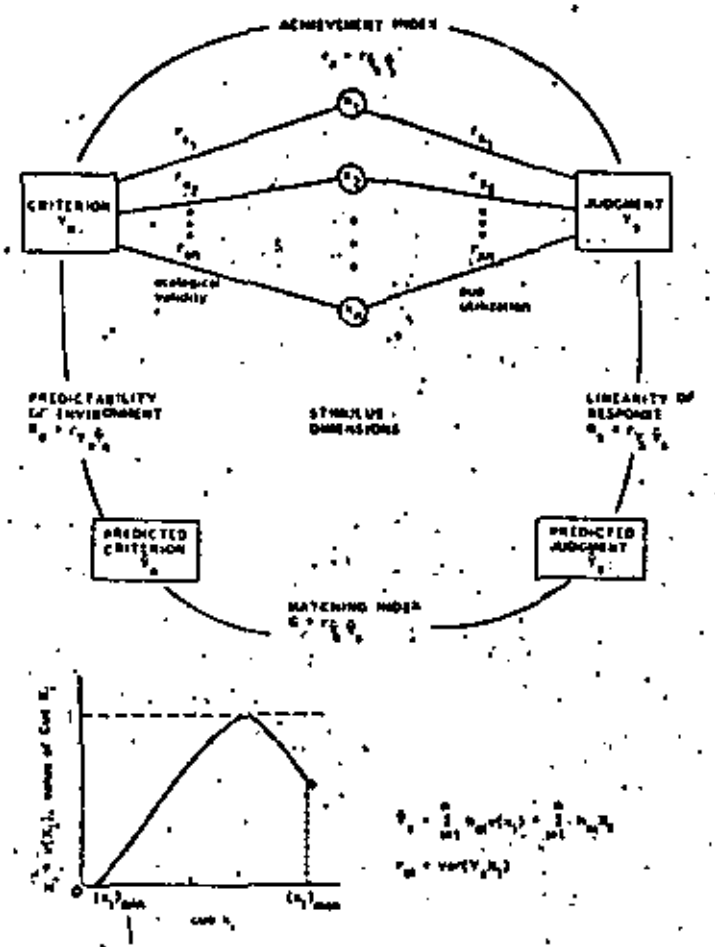


Fig. 3. The Brunswick lens model and its relation to Hammond's social judgment theory.

We have just described what might be regarded as a component or physiological model of information processing. In these stimulus response approaches, behavior is seen as being initiated by the onset of stimuli. A seeming deficiency in approaches of this sort is that there is little consideration of how information bits are aggregated to influence choice and how the decisionmaker goes about the process of information formulation or acquisition, analysis, and interpretation.

A lens model developed by Brunswick and his students is a notable exception to this. The Brunswick lens model is the basis for the policy capture or social judgement theory approach of Hammond and his colleagues [140]–[143]. The lens model, displayed in Fig. 3, assumes that people are guided by rational programs in their attempt to adapt to the environment. There is a criterion value Y_e, and the subjects response, judgment, or inference Y_s. The left side of Fig. 3 represents ecological cue validities which are the correlations r_{ei} between the cues and the criterion value. On the right or organismic side of Fig. 3, a subject will base a response, judgment, or inference Y_s, on the perceived ecological structure. By calculating the correlations r_{si} that exist between the cues and the response or criterion evaluation, learning concerning the response system can be obtained.

We note that the value of the environmental criterion Y_e and the subject inference Y_s are directly comparable if linear combinations of the cues are assumed. We have, for n cues,

$$Y_e = \sum_{i=1}^{n} b_{ei} x_i + r_{ei} \qquad \hat{Y}_e = \sum_{i=1}^{n} b_{ei} x_i$$

$$Y_s = \sum_{i=1}^{n} b_{si} x_i + r_{si} \qquad \hat{Y}_s = \sum_{i=1}^{n} b_{si} x_i$$

where b_{ri} and b_{ti} are optimum regression weights for the independent cues x_i which provide measures of the importance weights of the cues. r_e and r_s are error terms due to inadequacy of the linear model. Y_e and Y_s are the true criterion value and subject response, and $\hat{Y}_e$ and $\hat{Y}_s$ are the predicted criterion value and subject response based on the observed cues. The many works of Hammond and his associates [2], [21], [45], [54], [140]–[143], [186], [187], [261], [290], [294], [295], [404] concerning social judgment theory make use of this lens model. The approach has been shown to be useful in a variety of areas such as policy formulation, negotiation, and conflict resolution. Recent efforts by Hoffman, Earle, and Slovic [154] have shown that the computer displays of social judgment theory which show both task characteristics, in terms of cue values and corresponding criterion values; and response characteristics, in terms of individual cue values and associated subject responses and judgments; provide a very effective feedback mechanism which might enable people to effectively learn much about complex functional relationships and tasks. There are a number of studies of regression analysis approaches to determination of parameters for decision rules [260], [290]. Use of regression analysis is central to social judgment theory. Recent applications of the approach [261] have involved using simulation models to generate responses which are valuated by the decisionmaker.

Questions concerning the cognitive style used by the decisionmaking are, we believe, very important. Information analysis and information interpretation may be accomplished in a concrete operational mode of thought or in a formal operational mode. We will describe the essential features of these two higher-level cognition processes in Section V. The concrete operational thought process, which is typically applied in familiar situations which people perceive to be well structured, may involve efforts such as reasoning by analogy, or affect, or standard operating procedures. The formal operational thought process, typically applied in situations with which the problem solver is unfamiliar and inexperienced, may involve explicit use of quantitative or qualitative analytical thought.

In either of these modes or "styles" of thought or cognition, information acquisition, analysis, and interpretation may be quite flawed. Many recent studies emphasize the strong need for modeling problem solving behavior in a descriptive, or positive sense in order to detect possible flaws in information processing. Our discussions thus far in this section have been concerned with physiological models in which people have input and output mechanisms, a memory for information storage and retrieval, and a central processor for coordination and control. Here, we wish especially to underscore the need not only for physiological or stimulus–response models but especially for process tracing [72], [95–98], [235] models of information formulation, analysis, and interpretation as well as associated decisionmaking. Knowledge of the actual unaided process of problem solving or descriptive process tracing should serve as a useful guide to the design of information systems that avoid, or at least ameliorate the effects of, cognitive heuristics and biases. This involves requirements for a knowledge of the ways in which people apply strategies in order to reach judgments.

A large number of contemporary studies in cognitive psychology indicate that the attempts of people, including experts, to apply various intuitive strategies in order to acquire and analyze information for purposes such as prediction, forecasting, and planning are often flawed. Many studies have been conducted to describe and explain the way information is acquired and analyzed and the results of faulty acquisition and analysis. Generally the descriptive behavior of subjects in tasks involving information acquisition and analysis is compared to the normative results that would prevail if people followed an "optimal" procedure. There have been a number of recent discussions of cognitive biases from several perspectives [61], [62], [98], [142], [154], [156], [160], [161], [185], [234], [263], [304], [309], [346–349], [351], [352], [385], [386], [406–408]. The recent texts by Nisbett and Ross [264] and Hogarth [159] concerning the strategies and biases associated with judgment and choice are especially noteworthy. Among the cognitive biases that have been identified are several which affect information formulation or acquisition, information analysis, and interpretation. Among these biases, which are not independent, are the following.

1) Adjustment and Anchoring [345], [383]—Often a person finds that difficulty in problem solving is due not to the lack of data and information, but rather to the existence of excess data and information. In such situations, the person often resorts to heuristics which may reduce the mental efforts required to arrive at a solution. In using the anchoring and adjustment heuristic when confronted with a large amount of data, the person selects a particular datum, such as the mean, as an initial or starting point, or anchor, and then adjusts that value improperly in order to incorporate the rest of the data such as to result in flawed information analysis.

2) Availability [383], [385]—The decisionmaker uses only easily available information and ignores not easily available sources of significant information. An event is believed to occur frequently, that is with high probability if it is easy to recall similar events.

3) Base Rate [25], [291], [386]—The likelihood of occurrence of two events is often compared by contrasting the number of times the two events occur and ignoring the rate of occurrence of each event. This bias often occurs when the decisionmaker has concrete experience with one event but only statistical or abstract information on the other. Generally abstract information will be ignored at the expense of concrete information. A base rate determined primarily from concrete information may be called a causal base rate whereas that determined from abstract information is an incidental base rate. When information updates occur, this individuating information often is given much more weight than it deserves. It is much easier for individuating information to override incidental base rates than causal base rates.

4) Conservatism [210], [259], [345]—The failure to revise estimates as much as they should be revised, based on receipt of new significant information, is known as conservatism. This is related to data saturation and regression effects biases.

5) Data Presentation Context [161]—The impact of summarized data, for example, may be much greater than that of the same data presented in detailed, nonsummarized form. Also different scales may be used to considerably change the impact of the same data.

6) Data Saturation—People often reach premature conclusions on the basis of too small a sample of information while ignoring the rest of the data that is received later on, or stopping acquisition of data prematurely.

7) Desire for Self-Fulfilling Prophecies—The decisionmaker values a certain outcome, interpretation, or conclusion and acquires and analyzes only information that supports this conclusion. This is another form of selective perception.

8) Ease of Recall [205], [382], [383]—Data which can easily be recalled or assessed will affect perception of the likelihood of similar events occurring again. People typically weigh easily recalled data more in decisionmaking than those data which cannot easily be recalled.

9) Expectations [161], [235]—People often remember and attach higher validity to information which confirms their previously held beliefs and expectations than they do to disconfirming information. Thus the presence of large amounts of information makes it easier for one to selectively ignore disconfirming information such as to reach any conclusion and thereby prove anything that one desires to prove.

10) Fact–Value Confusion—Strongly held values may often be regarded and presented as facts. That type of information is sought which confirms or lends credibility to one's views and values. Information which contradicts one's views or values is ignored. This is related to wishful thinking in that both are forms of selective perception.

11) Fundamental Attribution Error (Success/Failure Error) [263], [264]—The decisionmaker associates success with personal inherent ability and associates failure with poor luck in chance events.

This is related to availability and representativeness.

12) Gamblers Fallacy—The decisionmaker falsely assumes that unexpected occurrence of a "run" of some events enhances the probability of occurrence of an event that has not occurred.

13) Habit—Familiarity with a particular rule for solving a problem may result in reutilization of the same procedure and selection of the same alternative when confronted with a similar type of problem and similar information. We choose an alternative because it has previously been acceptable for a perceived similar purpose or because of superstition.

14) Hindsight [112–114], [316]—People are often unable to think objectively if they receive information that an outcome has occurred and they are told to ignore this information. With hindsight, outcomes that have occurred seem to have been inevitable. We see relationships much more easily in hindsight than in foresight and find it easy to change our predictions after the fact to correspond to what we know has occurred.

15) Illusion of Control [209], [210]—A good outcome in a chance situation may well have resulted from a poor decision. The decisionmaker may assume a feeling of control over events that is not reasonable.

16) Illusion of Correlation [115], [383]—A mistaken belief that two events covary when they do not covary is known as the illusion of correlation.

17) Law of Small Numbers (see Kahneman and Tversky [235])—People are insufficiently sensitive to quality of evidence. They often express greater confidence in predictions based on small samples of data with nondisconfirming evidence than in much larger samples with minor disconfirming evidence. Sample size and reliability often have little influence on confidence.

18) Order Effects [161], [184]—The order in which information is presented affects information retention in memory. Typically the first piece of information presented (primacy effect) and the last presented (recency effect) assume undue importance in the mind of the decisionmaker.

19) Outcome Irrelevant Learning System [96], [97]—Use of an inferior processing or decision rule can lead to poor results and the decisionmaker can believe that these are good because of inability to evaluate the impacts of the choices not selected and the hypotheses not tested.

20) Overconfidence [114], [183], [216]—People generally ascribe more credibility to data than is warranted and hence overestimate the probability of success merely due to the presence of an abundance of data. The greater the amount of data, the more confident the person is in the accuracy of the data.

21) Redundancy—The more redundancy in the data, the more confidence people often have in their predictions, although this overconfidence is usually unwarranted.

22) Reference Effect [30], [383]—People normally perceive and evaluate stimuli in accordance with their present and past experiential level for the stimuli. They sense a reference level in accordance with past experience. Thus reactions to stimuli such as a comment from an associate, are interpreted favorably or unfavorably in accordance with our previous expectations and experiences. A reference point defines an operating point in the space of outcomes. Changes in perceptions due to changes in the reference point are called reference effects. These changes may not be based upon proper, statistically relevant computations.

23) Regression Effects [183], [383]—The largest observed values of observations are used without regressing towards the mean to consider the effects of noisy measurements. In effect this ignores uncertainties.

24) Representativeness [382], [383]—When making inference from data too much weight is given to results of small samples. As sample size is increased, the results of small samples are taken to be representative of the larger population. The "laws" of representativeness differ considerably from the laws of probability and violations of the conjunction rule $P(A \cap B) \leq P(A)$ are often observed.

25) Selective Perceptions [161]—People often seek only information that confirms their views and values. They disregard or ignore disconfirming evidence. Issues are structured on the basis of personal experience and wishful thinking. There are many illustrations of selective perception. One is "reading between the lines" such as, for example, to deny antecedent statements and, as a consequence, accept "if you don't promote me, I won't perform well" as following inferentially from "I will perform well if you promote me."

26) Spurious Cues [161]—Often cues appear only by occurrence of a low probability event but they are accepted by the decisionmaker as commonly occurring.

27) Wishful Thinking—The preference of the decisionmaker for particular outcomes and particular decisions can lead the decisionmaker to choose an alternative that the decisionmaker would like to have associated with a desirable outcome. This implies a confounding of facts and values and is a form of selective perception.

Doubtlessly there are other information acquisition, analysis, and interpretation biases that we have not identified here. Any categorization into acquisition, analysis, and interpretation bias is somewhat arbitrary since iteration and feedback will often, in practice, not allow this separation. Also, many of the identified biases overlap in meaning and, therefore, are related to others. Some further discussion of cognitive biases will be presented in our discussion of the situation framing phase of prospect theory in Section III. Certainty, reflection, and isolation effects are three results of these biases that have particular prominence in prospect theory.

Of particular interest are circumstances under which these biases occur, their effects on activities such as decisionmaking, issue resolution, planning, and forecasting and assessment; and appropriate styles which might result in debiasing or amelioration of the effects of cognitive bias.

Many of the cognitive biases that have been found to exist have been found in the unfamiliar surroundings of the experimental laboratory, and generalization of this work to real world situations is a contemporary research area of much interest. However most of the laboratory experiments have concerned very simple if unfamiliar tasks. A number of studies have compared unaided expert performance with simple quantitative models for judgment and decisionmaking, such as those by Brehmer [47], Cohen [62], Dawes [70], [71], Goldsmith [132], Kleinmuntz and Kleinmuntz [204], and by several authors in Wallsten's recent definitive work concerning cognitive processes in choice and decision behavior [396]. While there is controversy [62], [263], [349], most studies have shown that simple quantitative models perform better in human judgment and decisionmaking tasks, including information processing, than wholistic expert performance in similar tasks. This would appear to have major implications and to sound major caveats for such areas as "expert forecasting." This caution is strongly emphasized in the works of Hogarth and Makridakis [161], Makridakis and Wheelright [235], and Armstrong [14]–[16]. This is a caution noted in but a few [18] of the contemporary works on forecasting and assessment.

There are a number of prescriptions which might be given to encourage avoidance of possible cognitive biases and to debias those that do occur [96], [98], [161], [184], [235], [355], [386]. Some suggestions to avoid cognitive bias follow.

1) Sample information from a broad data base and be especially careful to include data bases which might contain disconfirming information.

2) Include sample size, confidence intervals, and other measures of information validity in addition to mean values.

3) Encourage use of models and quantitative aids to improve upon information analysis through proper aggregation of acquired information.

4) Avoid the hindsight bias by providing access to information at critical past times.

5) Encourage decisionmakers to distinguish good and bad decisions from good and bad outcomes in order to avoid various forms of selective perception such as, for example, the illusion of control.

6) Encourage effective learning from experience. Encourage understanding of the decision situation and methods and rules used in practice to process information and make decisions such as to avoid outcome irrelevant learning systems.

7) Use structured frameworks based on logical reasoning [255], [376] in order to avoid confusing facts and values and wishful thinking and to assist in processing information updates.

8) Both qualitative and quantitative data should be collected, and all data should be regarded with "appropriate" emphasis. None of the data should be overweighted or underweighted in accordance with personal views, beliefs, or values only.

9) People should be reminded, from time to time, concerning what type or size of sample from which data are being gathered, so as to avoid the representativeness bias.

10) Information should be presented in several orderings so as to avoid recency and primacy order effects, and the data presentation context and data saturation biases.

Kahneman and Tversky [235] discuss a systemic procedure to enhance debiasing of information processing activities. A definitive discussion of debiasing methods for hindsight and overconfidence is presented by Fischhoff [185]. Lichtenstein and Fischhoff present a number of helpful guidelines to assist in training for calibration [217]. Clearly more efforts along these lines are needed. Studies to determine the extent to which learning feedback acquired through use of methods such as social judgment theory contributes to debiasing would be especially rewarding. This is especially the case since confidence in unaided judgment is learned and maintained through feedback even when there is very little or no justification for this confidence [94]. Typically outcomes which follow from decisions based on negative judgments are not observed. Reinforcements of self-fulfilling prophecy type judgments through positive outcome feedback only occur in spite of, rather than due to, judgment validity.

Research integrating the methods whereby people integrate or aggregate information and attribute causes [8]–[12], [142], [143], [186], [190], [199], [321], [364] with methods for the identification and amelioration of cognitive biases would be of interest and of much potential use also.

In a sense, the results of this section are disturbing in that they tend to support the "intellectual cripple" hypothesis of Slovic ([142], pg. 14) and imply that humans may well be little more than masters of the art of self deception. On the other hand there is strong evidence that humans are very strongly motivated to understand, to cope with, and to improve themselves and the environment in which they function. While there are a number of fundamental limitations to systemic efforts to assist in bettering the quality of human judgment, choice, and decisions [282], [307], there are also a number of desirable activities [16], [305], [385]. These can assist in increasing the relevance of systemic approaches such as those which result in information processing adjuvants for policy analysis, forecasting, planning, and other judgment and decision tasks in which information acquisition, analysis, and interpretation play a needed and vital role.

IV. Decision Rules

In order to select an alternative plan or course of action for ultimate implementation, the decisionmaker applies one or more decision rules which enable comparison prioritization, and ultimately, selection of a single policy alternative from among a set of choice alternatives. The purpose of a decision rule is to specify the most preferred alternative generally from a partial or total ordering or prioritization of alternatives. To utilize a decision rule we must have a set of alternatives, a set of objectives to be accomplished by the alternatives, a knowledge of the impacts of the alternatives, evaluation of these impacts, and associated preference information. Decision rules may be explicit or implicit in terms of the way in which they are used in the decision process.

We can assume, without loss of generality, that each single policy alternative may represent a complex portfolio of individual alternatives and that the set of choice alternatives contains mutually exclusive components. This formulation can always be accomplished but may result in a very large set of policy alternatives since n individual alternatives can be combined into 2^n possible portfolios of alternatives. Failure to consider a combination of alternatives may result in significant errors in decisionmaking unless each of the individual alternatives represents one component of a portfolio of all possible combinations of individual alternatives, or unless the individual alternatives are independent or mutually exclusive.

It is assumed at the interpretation step of the decision process that formulation and analysis have been accomplished such that there exists a decision situation structural model and the results of exercising the model. Thus objectives, relevant constraints, some bounds on the issue, possible policy alternatives, impacts of policy alternatives, etc. are assumed known. The choice of a decision rule will depend, in large measure, upon the decision situation structural model as reflected in the contingency task structure. We will discuss dynamic models for contingency task structures in our next section.

The above discussion may appear representative primarily of the judgment and decision process associated with the formal operational thought mode that we will elaborate upon in our next section. For purposes of clarity of exposition, we have presented an oversimplified view of how decision rules are used to aggregate information and evaluate alternatives. The sequence we have described implies comparison and evaluation of alternatives only after we have first accomplished formulation and analysis of the issue under consideration. As we have noted throughout our discussion, decisionmakers typically compare and evaluate alternatives while they are in the process of decision situation formulation and analysis. These partial comparisons and evaluations lead to searches for additional policy alternatives, additional analysis, etc. As we have also noted, the entire decision process typically occurs in a parallel-simultaneous iterative fashion rather than an exclusively sequential series of steps in which formulation is followed by analysis which is followed by interpretation.

Individuals and decision environments vary so greatly that there are a great number of decision rules that will be needed to describe actual decision situations. Schoemaker [315] is among a number of authors [121], [255], [364], [365], [372] who have attempted classification schemes to allow categorization of various descriptive decision rule models. His first level categorization separates decision rules into holistic and nonholistic categories. In a holistic decision rule each alternative or portfolio of alternatives is evaluated and assigned a value or utility. After all alternatives have been evaluated, they are compared and alternative A is said to be preferred to alternative B if its evaluation has given it a greater utility such that $U(A) > U(B)$. In nonholistic decision rules individual alternatives or portfolios of alternatives are generally compared with one another in a sequential elimination process. This comparison may be against some standard across a few attributes within alternative pairs or across alternatives, with alternative attributes being compared one at a time.

Each of these categories appears to imply disaggregation into components of the event outcomes likely to follow from decisions. Our section on contingency task structure models will propose a dynamic evolving cognitive style model which admits of expert situational understanding that involves reasoning by analogy, intuitive affect, and other forms of nonverbal, almost unconscious perception. We elect to call this type of reasoning wholistic and add a third category to the classification scheme of Schoemaker.

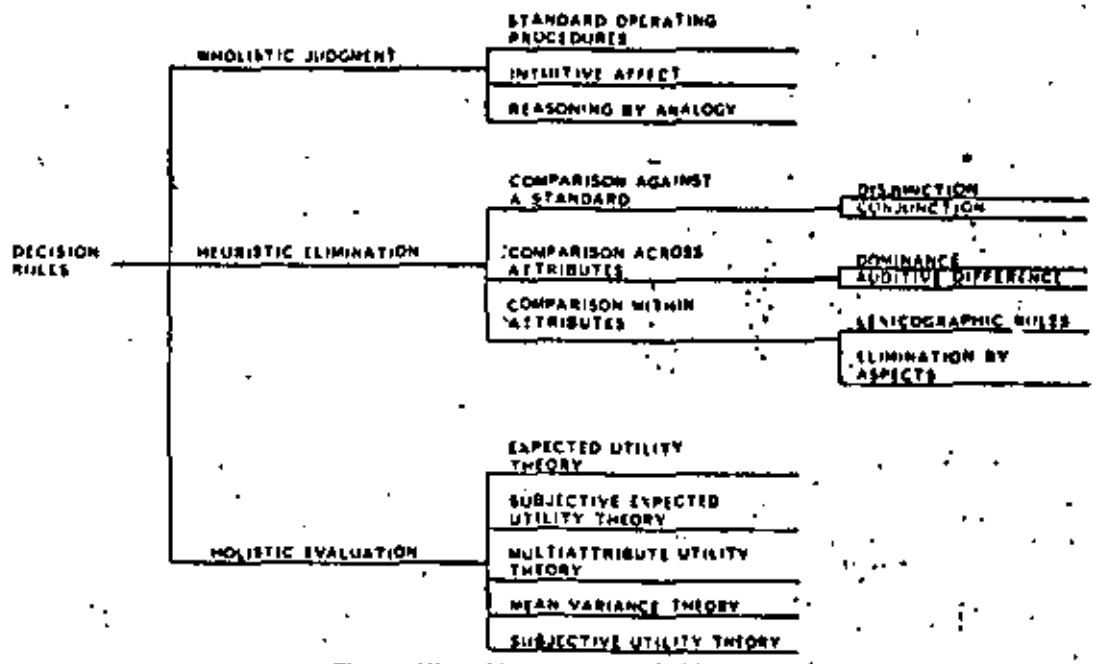


Fig. 4. Hierarchical structure of decision rules.

Consequently we envision three first level general categories of decision rules: holistic, heuristic, and wholistic. In a *holistic* decision rule there is an attempt to consider all aspects of a decision situation in evaluating choices by means of disaggregation of various choice components. In a *heuristic* decision rule, detailed complicated comparisons are not used. Rather, simplified approximations to holistic decision rules are used. In a *wholistic* decision rule, the evaluation and choice of alternatives is based upon use of previous experience, hopefully true expertise, with respect to similar decision situations. The selection of an alternative is based upon its perceived or presumed worth as a whole and without detailed conscious consideration of the individual aspects of each alternative. It is possible to define a number of decision rules and categorize them. The first level categories we have defined are not mutually exclusive. A number of decision rules doubtlessly can be categorized into more than one of these first-level decision categories. Figure 4 illustrates a possible inclusion structure for the decision rules we will describe here.

Expected Utility Theory: Our first decision rule is based on expected utility theory and is doubtlessly the most familiar decision rule to engineers. This rule derives from a "rational actor"[2] decision model [3], [4], [89], [103], [121], [134], [169], [192], [222], [256], [265], [285], [315], [359], [397] which is more fully discussed in Section VI.

The rational actor model is a normative model. Von Neuman and Morgenstern, who introduced the axioms of the model of rational man, stated the purpose of their work as

> to find mathematically complete principles which define "rational behavior"... a set of rules for each participant which tell him how to behave in every situation which may conceivably arise.

The idea of rationality originated in the economics literature where microeconomic models of the consumer and the firm assumed complete information and rationality. The rational person is assumed to have identified a set of well-defined objectives and goals and is assumed to be able to express preferences between different states of affairs according to the degree of satisfaction of attaining these objectives and goals. A rational person has identified available alternative courses of action and the possible consequences of each alternative. The rational person makes a consistent choice of alternative actions in order to maximize the expected degree of satisfaction associated with attaining identified objectives and goals.

A number of elements are assumed to exist in the rational actor model:

1) a set of policy alternatives A;
2) the set of possible consequences of choice or future states of nature or decision outcomes called S;
3) a utility function $U(s)$ that is defined for all elements s of S;
4) information as to which outcomes will occur if a particular policy alternative a in A is chosen; and
5) information as to the probability of occurrence of any particular outcome if an alternative $a \in A$ is chosen. $P_a(s)$ is the probability that $s \in S$ will occur if $a \in A$ is chosen.

There are a number of ways in which the axioms associated with the rational actor model may be stated. Each statement of the axioms allows proof of the fact that cardinal utility functions will exist and be unique only up to positive linear transformations. Further, the evaluation of expected utility allows choicemaking and prioritization of alternatives in accordance with the expected utility of each alternative. There are a number of textbook accounts of expected utility theory to which the interested reader of this review may turn for alternative sets of axioms and detailed accounts of the use of expected utility theory [51], [163], [196], [222], [285], [302], [315]. MacCrimmon and Larson interrelate the major axiom systems in expected utility theory [3] In a noteworthy contribution to understanding of the several systems that lead to (essentially) the same results for the rational actor model. A very readable introductory treatment of expected utility theory, relating descriptive psychological concerns with normative concerns, is presented by Vlek [244].

The rational actor model is often accepted as a normative model of how decisions should be made, at least in a substantive or "as if" fashion. It is often observed that the model is not an accurate description of either the substance or the process of actual unaided choicemaking behavior. Some of these observers use empirical evidence of the deviation of actual decisionmakers from either substantive rationality or process rationality. These observations are doubtlessly correct. The rational actor model is

[2] Technological or economic rationality would be a more appropriate term.

however, invaluable in that it can be often used as reference for comparison of actual behavior with ideal "aided" or normative behavior. Further, it provides a benchmark against which to compare simplified heuristics. Our efforts and discussions in this section concern primarily substantive behavior although we recognize the great difficulty, in practice, of separating substance from process.

Simon and his colleagues introduced the concept of bounded rationality and developed a satisficing model for individual choicemaking. It is worth noting that boundedly rational actors are basically rational subject to constraints on the formulation, analysis, and interpretation of information, and the substitution of achievement of a target level of return, or aspiration level, for selection of the best alternative. Typically people satisfice by adaptive adjustment [72] of aspirations such that, in repetitive decision situations, optimizing behavior is approached [270].

There is absolutely nothing in the formulation of the rational actor model which requires identification of *all* objectives, *all* possible alternatives, *all* possible impacts of alternatives, etc. The rational actor model is perfectly capable of being used to allow prioritization and selection of the best alternative by evaluating some impacts and with knowledge of some objectives, from among an incomplete set. It in no sense necessarily requires completeness in everything and the associated complexity that this would require. Actual decisionmaking behavior may not, however, even be boundedly rational but may employ such poor heuristics as to result in inferior choicemaking even to the extent of selecting inferior choices from among those in a bounded set.

There have been a number of experimental studies and field studies of the appropriateness of the expected utility model [3], [111], [117], [119], [125], [184]–[186], [236], [336]–[341], [385] as a descriptive model of substantive unaided behavior. Among the surveys which comment upon the experimental and field studies are [27], [98], [206], [348], [372]. Schoemaker [315] provides a very readable brief survey of some of this literature. While the evidence is mixed, most studies indicate that the expected utility decision rule simply does not function well in a *descriptive substantive* sense.

In its simplest form the expected utility of alternative a_i is computed from

$$E[U(a_i)] = \sum_{j=1}^{n} p[s_j(a_i)]U[s_j(a_i)] \qquad (1)$$

where the $s_j(a_i)$, $j = 1, 2, \ldots, n$, are the states which may result from alternative a_i and the $p[s_j(a_i)]$ are the associated probabilities. In the expected utility formulation the $p[s_j(a_i)] = p_j(a_i) = p_j$ are assumed to be objective probabilities and, of course, $\sum_{j=1}^{n} p_j = 1$. Generally these probabilities are not alternative invariant although notationally they are sometimes written as if they were independent of alternatives. The $U[s_j(a_i)]$ are the utilities or values [296] of the decisionmaker for the various outcome states. Johnson and Huber [179] survey a number of procedures that can be used to elicit utility functions. Most of the textbooks cited earlier also contain discussion of utility assessment procedures.

Subjective Expected Utility: Often it occurs that objective probabilities are, for any of a variety of reasons, unavailable in a given situation. The subjective expected utility model is obtained when subjective probabilities $f(p_j)$ are substituted for the p_j in (1). The $f(p_j)$ are generally elicited such that $\sum_{j=1}^{n} f(p_j) = 1$ and so the subjective probabilities behave in a way consistent with the laws of probability. There are a number of discussions concerning probability elicitation [31], [223], [257], [355] that present appropriate procedures to enable determination of subjective probabilities from individuals and groups. Conventional approaches to elicitation of utility in expected utility theory may confound strength of preference felt for alternative event outcomes and attitude toward risk. Also the elicitation procedure can become cumbersome. Recent research has formally separated these factors [33] and shows much promise in enhancing understanding of attitude towards risk. In this approach the utility concept is devoid of risk. It takes on a meaning more like that in conventional microeconomics where it measures strength of preference for certain outcomes only. This research [33] could provide additional linkages and understanding between the expected utility and subjective expected utility concepts by providing for incorporation of risk aversion effects in a relatively simple way. A related approach to incorporation of risk aversion is described by Howard and decision analysts at the Stanford Research Institute [164] who have been responsible for a number of major application studies in this area. There have been a number of related approaches [65], [66], [121] and the subjects of risk and uncertainty are of much contemporary interest [6], [136], [153], [304].

A number of studies have indicated that the relation between subjective and objective probabilities is nonlinear and situation dependent. It is usually indicated that people often underestimate high probabilities and overestimate low ones. More recent research has indicated that this appears true only for favorable outcomes. Just the opposite appears true when the outcome is unfavorable. This appears to be a form of wishful thinking for low probability events and "everything bad happens to me" for high probabilities. What we will call subjective utility theory attempts to incorporate situation dependent nonlinearities that may exist between subjective and objective probabilities.

Multiattribute Outcomes: Often decision situations are sufficiently complex that it is difficult to evaluate, in a wholistic fashion, the utility of each outcome. Often it is possible to disaggregate the features on which utility is based into a number of components called attributes. An attribute tree is a hierarchical structure which, when quantified through elicitation of values of the outcomes on the lowest level attributes and relative weights of the attributes, can be used to determine the utility of event outcomes. The types of multiattribute utility models used have varied from very simple unit weight linear models to rather complex multiplicative models [106]. Dawes [71] documents the robust beauty of linear models of the form

$$U(s_i) = \sum_{j=1}^{m} h_j u_j(s_i), \quad \sum_{j=1}^{m} h_j = 1 \qquad (2)$$

where there are assumed to be m attributes, h_j is the weight of the jth attribute and $u_j(s_i)$ the value score on the jth attribute of outcome s_i. In much of the work in this area decisions under certainty are considered such that there is a one to one correspondence between alternative a_i and outcome s_i. Under decision-under-certainty conditions we can let $s_i = a_i$ in (2).

Multiattribute models have been very successfully used to predict the decision behavior in field settings of many professional groups. Hammond [140]–[142] and his colleagues have, as discussed in Section III, developed an approach known as social judgment theory in which the "policy" of the decisionmaker, equivalent in this circumstance to the weights h_j, are identified from wholistic prioritization of decision outcomes through use of regression analysis techniques. Ward Edwards and his colleagues [186], [301] and elsewhere, elicit weights from decisionmakers for the model of (2) in a useful straightforward procedure called simple multiple attribute ranking technique (SMART) that has seen a number of realistic applications. Results of the surveys of Armstrong [14], [15]; Fischer [111]; Slovic and Lichtenstein [345]; Slovic, Fischhoff, and Lichtenstein [348]; Shanteau [324], and others indicate that simple linear models [64] are very potent predictors of reliable judgment, especially under conditions of certainty in that one can replicate the substantive judgment of decisionmakers. This is the case even though the simple linear model may not do a very good job of modeling the decision process. "Boot strapping" is the name given to the task of substituting a decision rule for the decisionmaker. The studies in the cited references show that the elimination of human judgment

error made possible by boot strapping enables it to be superior to unaided human judgment. One can even misspecify weights and ignore attribute dependencies and still find that weighted linear models do quite well [71].

The fact that the weighted linear rule may be so good is a rather mixed blessing. In circumstances in which there is no requirement for knowledge of the underlying decision process, the substantive predictive ability of the linear additive model may make it quite useful. Situations such as evaluating credit card applicants or applicants for admissions to colleges are repetitive judgment and decision situations which fit into this category. Use of simple formal linear model may well, in situations such as these, lead to a more efficient as well as more effective and equitable selection process than one based on unaided human intuition [70], [71]. (Dawes in Shweder [332]). In unstructured or semistructured nonrepetitive decision situations it is much less clear that a decision rule that is not guaranteed to be faithful to the underlying decision process will be nearly as valuable as one that is in terms of enabling decisionmakers to make better decisions. Fischhoff, Goitein, and Shipira [119] provide a number of perceptive comments concerning this and the consequent need for a theory of errors to explicate the effects of poor decision situation structural models and parameters within the structure. A hoped-for achievement is a sensitivity-based analysis of deviations from optimality to determine, among other things, the role of experience in decisionmaking and those components and principles of decisionmaking which can be usefully and meaningfully learned from experience [47], [94]–[97], [113], [116].

Multiattribute utility models based on the expected utility theory of von Neumann and Morgenstern are considerably more complex than those of behavioral decision theory. Often there are efforts to determine existence of various attribute independence conditions such as to validate use of a linear model of the form of (2) or a multiplicative model of the form

$$1 + kU(x_i) = \prod_{j=1}^{m} \left[1 + k_j u_j(x_i)\right], \qquad \sum_{j=1}^{n} k_j = 1. \quad (3)$$

The foremost proponents of this approach are Keeney and Raiffa [196]. There are many contributions to this area and variations of the basic approach [23], [29], [75], [93], [127], [231], [277], [278], [300], [301], [302], [358], [399]. It is proposed exclusively as a normative approach and has been successfully used for a variety of applications including proposal evaluation [245], [310], siting power plants [197]; and budgeting and planning [52], [191].

Mean Variance—There are a number of models and associated decision rules based upon mean-variance (EV) models. Markowitz's portfolio theory, which is well summarized in Libby and Fishburn [214] and Baron [26], is based in part on the assumption of a quadratic utility function

$$U(x) = \alpha + \beta x + \gamma x^2, \quad (4)$$

where the same states are assumed invariant over all alternatives such that we have a quadratic programming problem in prioritizing alternatives where

$$\begin{aligned} E[U(a_i)] &= \sum_{j=1}^{n} p_j(a_i)U(x_j) \\ &= \alpha + \beta E(a_i) + \gamma E(a_i^2) \\ &= \alpha + \beta\mu_i + \gamma(\sigma_i^2 + \mu_i^2). \end{aligned}$$

Coombs [65], [66], [185] has also been concerned with portfolio theory and assumes an optimum risk level in the form of a single peaked risk preference function for every expected value level. Gambles of equal expected value are judged on the basis of lower variance in the Markowitz portfolio theory and on the basis of deviation from optimum risk level in Coombs' portfolio theory. Stochastic dominance concepts [124] are especially useful in dealing with problems in the mean–variance models of portfolio theory. Unfortunately as has been shown by a number of authors [124], the results from using mean–variance portfolio theory are not necessarily consistent with results obtained from expected utility theory. For example, if the outcomes of decision a_1 are \$10 with probability 0.5 and \$20 with probability 0.5 and the outcome of decision a_2 is \$10 with probability 1.0; then the EV rule ($\mu_{a1} = \$15$, $\sigma_{a1} = \$5$) ($\mu_{a2} = \$10$, $\sigma_{a2} = 0$) is indeterminate in that there is no pareto superior or dominance alternative in an EV sense. Yet any reasonable person would prefer alternative a_1 to alternative a_2.

Fishburn [123] has considered a variation of the mean–variance model which involves concepts based upon target level of return or aspiration level or reference level to define the risk of an alternative. The "risk" of alternative a is determined from the probability of receiving a return not to exceed x, denoted $F(x)$, by

$$R(a) = \int_{-\infty}^{t} (t - x)^{\alpha} dF(x) = \int_{-\infty}^{t} (t - x)^{\alpha} p(x)\, dx$$

where t is the target return, α is a nonnegative parameter that is used to indicate relative importance of deviations below target return. For $0 \leq \alpha < 1$ the decisionmaker's primary concern is failure to achieve the target with little regard to the size of the deviation. For $\alpha > 1$ the decisionmaker is very concerned with sizeable deviations from target and relatively unconcerned with small deviations. In the former case the decisionmaker is risk seeking for losses and has a utility function that is convex for losses. In the latter case the decisionmaker is risk averse for losses and has a utility function that is concave for losses.

In this model the mean return from an alternative and its risk are the two attributes determining preference. This model thus appears much similar to the standard EV model in that $a_1 > a_2$ if and only if $\mu(a_1) \geq \mu(a_2)$ and $R(a_1) \leq R(a_2)$ with at least one inequality being valid. In the example just considered, the mean values are as given previously and the risks are

$$R(a_1) = \begin{cases} 0, & t \leq 10 \\ 0.5(t - 10)^{\alpha}, & 10 \leq t \leq 20 \\ 0.5(t - 10)^{\alpha} + 0.5(t - 20)^{\alpha}, & 20 \leq t \end{cases}$$

$$R(a_2) = \begin{cases} 0, & t \leq 10 \\ (t - 10)^{\alpha}, & 10 \leq t \end{cases}$$

Thus we see that the risk is the same, that is zero, if $t \leq 10$ and so we prefer a_1. The risk associated with a_1 is one half that associated with a_2 if the target return is between \$10 and \$20. The risk associated with a_1 is less than that associated with a_2 if $t \geq 20$. And so, since $\mu(a_1) > \mu(a_2)$, we prefer a_1 regardless of the target return. Generally, as in this case, Fishburn's below-target model will resolve ambiguities associated with the standard mean-variance model. The decisionmaker is free to specify α and t. Thus this represents a rather useful dominance type decision rule. Extensions of this rule to the case of multiattribute and multiple objective preferences would have considerable value.

Subjective Utility Theory: A number of researchers have proposed holistic decision rules based on the observation that people, in unaided situations, do not typically perceive (objective) probabilities such that the fundamental probability property $\sum_{j=1}^{n} p_j = 1$ is satisfied. There presently exists several decision situation models based upon a subjective utility theory in which probabilities do not sum to one. Among these are certainty equivalence theory due to Handa [144], subjectively weighted utility theory due to Karmarkar [188], [189]; and prospect theory due to Tversky and Kahneman [184], [385]. There have been several additional studies involving prospect theory including those of Thaler [371], and Hershey and Schoemaker [152], [153]. Some of the foundations for these subjective utility theory efforts can be found in the early work of Allais [3] who was among the

first to note that the normative expected utility approach of von Neumann and Morgenstern, and the subjective expected utility modifications, did not necessarily describe actual descriptive choice behavior. We believe that these studies are especially relevant to information system design and so summarize relevant features from these effects here.

In certainty equivalence theory five axioms are assumed. We will use the term prospect or prospect (s, P) to mean the opportunity to obtain outcome s with probability P. Simply stated, these are as follows.

1) Preferences are governed only by utilities and outcomes. One is indifferent between a nonsimple prospect and an actuarially identical simple prospect with a single event node.
2) Complete ordering of prospects is possible and transitivity of prospects exists.
3) Continuity exists such that if $(s_1, P_1) \succcurlyeq (s_2, P_2), \succcurlyeq (s_3, P_3)$ then there exists an α such that $(s_2, P_2) \sim [\alpha s_1, P_1) + (s_3 - \alpha s_3, P_3)$.
4) Independence exists such that if $(x_i, P_i) \sim (x_i, 1) \forall i$, then $(\underline{x}, \underline{P}) \sim (\Sigma x_i, 1)$ where $\underline{x}$ and $\underline{P}$ represent vectors of outcomes and probabilities x_i and P_i.
5) Enhanced prospects are preferred if and only if a basic prospect is preferred. Thus $(\beta s_1, P_1) \succcurlyeq (\beta s_2, P_2) \forall \beta \geq 0$ if and only if $(s_1, P_1) \succcurlyeq (s_2, P_2)$.

These axioms are sufficient to insure that the subjective utility function of alternative a_i, $CE(a_i) = CE[s(a_i), P(a_i)] = U(\underline{s}^i, \underline{P}^i)$, is linear in s_i and of the form

$$U(\underline{s}^i, \underline{p}^i) = \sum_{j=1}^{n} s_j^i w(P_j^i) = \underline{w}^T(\underline{P}^i)\underline{s}^i. \tag{5}$$

Axioms 1, 4, and 5 incorporate the major changes from the von Neumann–Morgenstern axioms. It appears unduly restrictive to require that the utility function be linear in the outcome and this is reason enough to warrant the development of a more robust theory.

Fishburn [125], however, has shown that certainty equivalence theory must reduce to the expected value model, $U(\underline{s}, \underline{P}) = \underline{P}^T\underline{s}$, $\Sigma_{j=1}^{n} w(P_j) = 1$. This occurs because of the requirement that one must be indifferent between a nonsimple prospect and an actuarially equivalent simple prospect. To insure this for the two outcome case, for the general actuarially equivalent two outcome prospects of Fig. 5, requires that $w(P) + w(1 - P) = 1$.[3] This certainty must be viewed as another limitation of this certainty equivalence theory and indicates the considerable care that must be exercised in modifying the basic utility theory axioms.

The subjective weighted utility model yields for the SWU of alternative a_i,

$$SWU(a_i) = \sum_{j=1}^{n} w[P_j(a_i)]U[s_j(a_i)] \tag{6}$$

where the subjective weighted probabilities are

$$w[P_j(a)] = \frac{f[P_j(a)]}{\sum_{j=1}^{n} f[P_j(a)]} \tag{7}$$

Although a variety of probability weighting functions are possible, Karmarkar [188], [189] proposes use of a log normal function

$$\ln\left(\frac{f}{1-f}\right) = \alpha \ln\left(\frac{P}{1-P}\right) \tag{8}$$

[3] For the n outcome case we would have $\Sigma_{j=1}^{n} w(P_j) = 1$ and we see that the only general $w(P_j)$ that will insure this is $w(P_j) = P_j$.

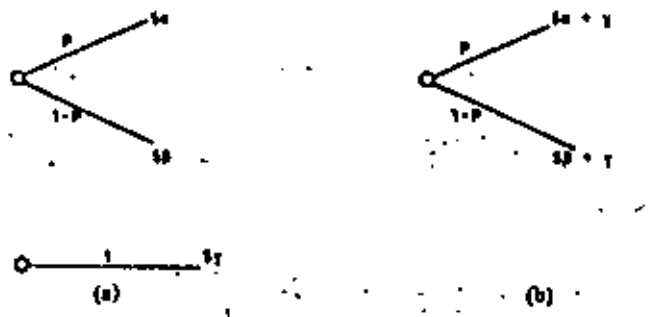


Fig. 5. Two actuarially equivalent prospects.

or

$$f(P) = \frac{P^\alpha}{P^\alpha + (1-P)^\alpha} \tag{9}$$

where $0 \leq \alpha \leq 1$. This transformation of probabilities is such that large probabilities are understated and small probabilities overstated. Karmarkar emphasizes that the probability weighting function does not represent a probability perception phenomenon but represents a bias in the way in which (objective) probabilities are descriptively incorporated into the evaluation, prioritization, and choice making process. In this model, the final weighted probabilities do sum to one in accordance with the conventional subjective expected utility theory. However, the expression for any nonlinear weighted probability $w[P_j(a)]$ is actually a function of the value of all other probabilities as seen in (7). The effects of this confounding influence remain to be investigated.

The considerably more sophisticated prospect theory of Tversky and Kahneman [384], [385], contains a number of modifications to expected utility theory. Prospect theory consists of an editing phase involving framing of contingencies, alternatives, and outcomes, followed by an evaluation phase. These modify subjective expected utility theory such as to enhance bounded descriptive realism of the theory.

1) In the editing phase the decision situation is recast into a number of simpler situations in order to make the evaluation task simpler for the choicemaker. The tasks in editing are very much dependent on the contingency situation at hand and offer possibilities for coding, combining, segregating, cancelling, and detection of dominance.
2) Value functions are devoid of risk attitude and are unique only up to positive ratio transformations.
3) Outcomes are expressed as positive or negative deviations from a reference or nominal outcome which is assigned a value of zero. Thus value changes represent changes in asset position. Positive and negative values are treated differently with the typical value function being an S-shaped curve that is convex below the reference point and concave above it. Displeasure with loss is typically greater than pleasure associated with the same gain.
4) Probability weights $w[P_j(a)]$ reflect an uncertain outcome contribution to the attractiveness of a prospect. As in SWU theory, high probabilities are underweighted and low ones overweighted. The following are among the properties of the probability weighting function.
   a) True at extremes $w(0) = 0$, $w(1) = 1$
   b) Subadditive at low P, $w(aP) > aw(P), 0 < a$
   c) Overweighted for small p, $w(p) > p$, $p \ll 1$
   d) Underweighted for large P, $w(P) < P$, $P > 0$
   e) Subcertain $w(p) + w(1-p) < 1$
   f) Subproportional $w(\alpha P)/w(P) < w(\alpha\beta P)/w(\beta P)$, $0 < \alpha, \beta, < 1$.
5) The value of a prospect $(\underline{s}, \underline{P}) = (s_1, P_1) + (s_2, P_2)$ is given by
   a) $V(\underline{s}, \underline{P}) = v(s_2) + w(P_1)[v(s_1) - v(s_2)]$ (10)

for strictly positive prospects in which $P_1 + P_2 = 1$ and $x_1 > x_2 > 0$, or strictly negative prospects in which $P_1 + P_2 = 1$, $x_1 < x_2 < 0$.

b) $V(\underline{x}, \underline{P}) = w(P_1)v(S_1) + w(P_2)v(x_2)$ (11)

for regular prospects which are prospects that are neither strictly positive nor strictly negative in that either $P_1 + P_2 \neq 1$ and/or $v(x_1)$ and $v(x_2)$ are of opposite sign."

In no sense is prospect theory posed as a normative theory of how people should make decisions. The editing or framing of contingencies, alternative acts, and outcomes is similar to the formulation step of the systems process. It is in this forming phase that the contingency task structure and decision situation model are, in effect, formed. For example, in a population of one million people where black lung disease might kill two thousand people, possible forms are

Form 1—Alternative a_1 will save 500 people whereas if alternative a_2 is adopted there is a 0.25 probability of saving 2000 people and a 0.75 probability of not saving anyone.

Form 2—Alternative a_1 will result in death of 1500 people whereas alternative a_2 will result in a 0.25 probability that no one will die and a 0.75 probability that 2000 people will die.

These two forms are really the same, yet many people will interpret them differently. The editing or forming phase of prospect theory allows different interpretations and thus makes provision for different evaluation of results in terms of alternative formulations of the same issue.

Prospect theory is especially able to cope with *certainty effects* in which people overweight outcomes considered certain compared with those considered only highly probable; *reflection effects* in which preferences are reversed when two positively valued outcomes are replaced by two negatively valued outcomes; and *isolation effects* in which people disregard common outcome components shared by outcomes and focus only on components that distinguish alternatives. Kahneman and Tversky have established an axiomatic basis for prospect theory [184] for the two outcome case.

In a recent study involving prospect theory, Hershey and Schoemaker [152] question the generality of the reflection hypothesis of prospect theory which states that asymmetric preferences are found when comparing gain prospects with loss prospects. They introduce four types of reflectivity depending upon whether subjects choose positive prospect (x_1, P_1) or the noninferior prospect (x_2, P_2), and whether they choose negative prospect $(-x_1, P_1)$ or $(-x_2, P_2)$. Across-subject and within-subject reflectivity are examined in terms of whether subjects do or do not choose and do or do not switch from safe to risky prospects. They conclude that predictions of prospect theory concerning reflectivity depend upon the size of probabilities. For P large enough to insure underweighting of probabilities, it appears that the reflectivity hypothesis is quite valid. For smaller values of P, reflectivity is neither predicted nor excluded from the results of Hershey and Schoemaker.

In another study Hershey and Schoemaker [153] examine preferences for basic insurance-loss lotteries and show that risk taking is prevalent in the domain of losses. They suggest a utility function which is concave for low losses and convex for larger ones. They indicate a context effect in which various insurance formulations lead to more risk averse behavior than for statistically equivalent gambling formulations. Their conclusion that probabilities and outcomes may be of less guidance in influencing decision behavior as uncertainties concerning their magnitude increase, strengthens conjectures concerning the influence of context and perceptions of decision situation structural models upon decision results.

Thaler [371] examines a number of the tenets of prospect theory with generally very positive confirming results. Additional comments concerning the seminal prospect theory appear in a previous survey in these transactions [304] including the observation that a number of the results of prospect theory, which are seemingly at variance with expected utility theory, can be accommodated successfully using multiple attribute utility theory. Extensions of prospect theory to include multiple attribute preferences, large number of outcomes, sequential multistage decisionmaking, risk aversion coefficients, and subjective probability effects would do much to enable this significant development to be of even greater usefulness in explaining complex positive or descriptive decision behavior. This might well be of much normative use as well.

Heuristic Decision Rules

A number of decision rules do not involve comparisons in a true holistic fashion. Rather they involve comparisons of one alternative with another, generally within a restricted alternative set and attribute set. Within the heuristic class of decision rules we may distinguish those which compare alternatives against some standard by means of conjunctive or disjunctive comparisons, those which compare alternatives across attributes, and those which make comparisons within attributes. Generally, these rules can result, when improperly applied, in intransitive choices [289]. We will consider several rules from each subcategory. First we will discuss two noncompensatory rules [90] that are often used when there is an overabundance of data present.

Disjunctive—A disjunctive decision rule is one in which the decisionmaker identifies minimally acceptable value standards for each relevant attribute. Alternatives which pass the critical standard on one or more attributes are retained. Alternatives which fall below the critical standards on all attributes are eliminated. A single alternative is accepted when the critical standards are set such that all but one alternative fail to exceed any of the critical standards on any attributes. Unlike multiattribute utility theory (MAUT) rules, where poor performance on one attribute can be made up by good performance on other attributes such that the rule is compensatory, a disjunctive decision rule is noncompensatory. A compensatory approximation to a disjunctive decision rule for attributes x_i is

$$U = \sum_{i=1}^{m} \frac{1}{\left(1 + \frac{x_i}{c_i}\right)^{n_i}}, \quad n_i \gg 1 \tag{12}$$

where m represents the number of attributes and c_i is the critical value on the ith attribute. If U is greater than one, the alternative in question is retained.

Conjunctive—A conjunctive decision rule is one in which minimally acceptable value standards for each relevant attribute are identified. Alternatives are acceptable if they exceed all minimum standards. They are rejected if they fail to exceed any minimum standard. The critical values for disjunctive and conjunctive rules are generally different. A compensatory approximation to the noncompensatory conjunctive decision rule is

$$U = \prod_{i=1}^{m} \frac{1}{\left(1 + \frac{c_i}{x_i}\right)^{n_i}}, \quad n_i \gg 1. \tag{13}$$

An alternative is retained if the corresponding utility U is above a threshold which is set just slightly below 1. These approximations for the disjunctive and conjunctive rules become noncompensatory as n_i approaches infinity.

By iterating through the conjunctive acceptance and disjunctive rejection rule several times with adjustable critical values or

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

DESIGN OF INFORMATION SYSTEMS AND PROCESSES

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984

aspiration levels, these rules become, in effect, forms of satisficing rules.

Dominance models and additive difference models are two examples of models which lead to decision rules involving comparison across some, but not necessarily all, attributes. No minimum standard of performance on attributes, that is to say minimum aspects, are identified.

Dominance—A dominance decision rule is one which chooses alternative a_1 over a_2 if a_1 is better than a_2 on at least one aspect and not worse than a_2 on any other aspect. An aspect is the score of a specific attention on a specific attribute. There are a number of applications of dominance theory, including stochastic dominance, to decisionmaking situations [33], [54], [75], [124], [358], [399].

Additive difference—In an additive difference rule [382]–[385], a binary choice is made between alternatives a_1 and a_2. Differences are considered between values for a_1 and a_2 on each relevant attribute. Differences of the form $U_i(a_1) - U_i(a_2)$ are computed. Each of the differences is weighted in proportion to the importance of the differences between alternatives on the various attributes. The resulting weight is $f_i[U_i(a_1) - U_i(a_2)]$. Alternative 1 is preferred to alternative 2 only if

$$\sum_{i=1}^{n} f_i[U_i(a_1) - U_i(a_2)] > 0.$$

This is a compensatory rule and can be used to compare any number of alternatives merely by retaining the winner in each comparison [272]. Only if the functions f_i are linear will the additive difference rule necessarily lead to transitive choices.

A third important subcategorization involves comparison within attributes. There are a variety of lexicographic procedures [121] and the elimination by aspects rule [381], [382] which explicitly involve comparison of alternatives on one, or at most a few, attributes.

Lexicographic Decision Rule: This rule prescribes a choice of the alternative which is most attractive on the most important attribute. If two aspects on this attribute are equally attractive, the decision will be based upon the most attractive aspect on the attribute next in order of importance, etc.

Minimum Difference Lexicographic Rule: This rule is much like the lexicographic rule, with the additional assumption that for each attribute there is a minimum acceptable difference Δ_i of alternative scores. Thus, only differences greater than Δ_i between the attractiveness values of two alternatives may determine a decision. If the difference on the most important attribute is less than Δ_1, then the attribute next in the lexicographic order is considered. The lexicographic semiorder rule is a special case of this decision rule where Δ_i is defined only for the most important attribute. For all other attributes $\Delta_i = 0$. This procedure may easily be extended to cases where the Δ_i are defined for the two most important aspects. This rule is often used in situations where information about attributes are missing as a result of imperfect discrimination among alternatives on a given attribute or of unreliability of available information. In general this rule leads to intransitive choices when there are more than two alternatives. It may even lead to agenda dependent results for the case where there are only three alternatives. One should be especially careful to examine relations used for ordering alternatives to attempt to detect use of heuristics such as this, especially if concepts such as transitivity are used, perhaps inferentially, to determine partial orderings. This suggests the need for special care when attempting to use transitivity concepts to infer ordinal preferences. The resulting failure to seek disconfirming information may well create structural preference illusions.

Einhorn [96], [97] first uses the term "outcome irrelevant learning structure" to describe processes which used deficient heuristics, and which then reinforces poor choices through experiences involving feedback and lack of disconfirming evidence. These outcome irrelevant learning structures (OILS) may result either from unaided judgment processes or from poorly conceived or possibly well conceived but improperly utilized, and therefore irrelevant, systemic methods or processes.

The Maximizing Number of Attributes in Greater Attractiveness Rule: This rule prescribes a choice of the alternative that has the greater number of favorable attributes. Specifically the rule requires that the aspect of a decision alternative must be classified for each attribute as better, equal, or worse than the attractiveness of the other alternative on that attribute. The preferred alternative will be that which has the greatest number of favorable classifications.

Elimination by Aspects: In this rule [288], [381] attributes are assumed to have different importance weights. An attribute is selected with which to compare alternatives with a probability that is proportional to its weight. Alternatives which do not have attribute scores above some aspiration or critical level are eliminated. A second attribute is selected with probability proportional to its weight and evaluation by elimination continues. The elimination by aspects model is thus seen to be a lexicographic rule in which decision-forming attributes are picked according to a probabilistic mechanism.

Wholistic Decision Rules

It is not possible to provide anywhere near a complete listing or discussion of the many possible wholistic decision rules. Three of these wholistic judgment processes occur perhaps more frequently than others: standard operating procedures, intuitive affect, and reasoning by analogy.

Standard Operating Procedures: Standard operating procedures may result from the application of holistic or heuristic procedures, or other wholistic judgment approaches. A standard operating procedure is essentially what the name implies, a set of experience-based guides to behavior which are typically used without resort to the underlying rationale which led to the procedure. Often standard operating procedures are formulated by one person or group and then implemented by another person or group. Sometimes they involve habit or folk custom, such as "drink white wine with fish."

Often user's guides and operating manuals are written in attempts to standardize operating procedures for performance. The greatest value of these procedures is as a checklist, reminder, or options, profile of attributes to look for, judgments to make or activities to select or perform. A fundamental often occurring difficulty is that an expert may be able to use a checklist or profile of options as a guide to performance based upon the ability of the expert to quickly recognize the features inherent in the situation. Lack of training and experience will often make it not possible for the novice to utilize this capacity for task need recognition. Klein and Weitzenfeld [202], [230] pose that the lack of training and experience inherent in the novice, the associated lack of ability to recognize contextual relations and analogous situations, and the inability of guides to be able to teach this ability are all fundamental impediments to the use of many standard operating procedure type guides to judgment and performance.

Intuitive Affect

A person who makes judgments based on intuitive affect typically takes in information by looking at the "whole" of a situation rather than by disaggregating the situation into its component parts and acquiring data on the parts. Valuation is typically based on an attempt to determine whether alternatives are pleasant or unpleasant, likeable or unlikeable, good or bad for individuals. It stressed the uniqueness of personalistic value judgments. Zajonc [411] presents a very useful discussion of affect or feeling as postcognitive activity.

Reasoning by Analogy

Many philosophers of science claim reasoning by analogy [55], [130], [360] is the basis of hypothesis generation. It is fundamentally different than deductive inference or inductive inference-based reasoning. In analogic reasoning we use analogies, prototypes, or other paradigms with which we are familiar to guide us in new tasks. These exemplars encourage recognition in a present situation in terms of experientially based knowledge.

Doubtlessly analogic reasoning, as well as reasoning by intuitive affect and standard operating procedures, are each heavily influenced by the contingency structure of the task at hand and the environment. These are the judgment processes used by many in reaching decisions. We will comment further in Sections V and VI upon wholistic judgment and its role [81], [82], [98], [116]-[119], [202], [203] in choicemaking.

In this section we have examined a number of decision rules. We have discussed holistic, heuristic, and wholistic rules. The holistic models or rules are generally substantive and not necessarily process models. They may be prescriptive or descriptive in intent and use. The heuristic and wholistic models are more process oriented than the heuristic models. In unaided situations people generally do not have the cognitive stamina to utilize the holistic rules or may not sense a need for them even if they could utilize them. A variety of contemporary research [272]-[275] has presented the strongest of evidence that choice of decision rules is very task dependent and actual choices may vary appreciably across different interpretations of the same decision situations. Preference reversals have even been noted with translation of gambles and target return, reference point, or aspiration level effects. Phenomena such as these have recently been studied [274] and shown to be potentially explainable by a descriptive model of risky choice due to Fishburn [123] and by prospect theory.

We note that people use different decision rules and models at different phases of a decision process as a function of a number of influencing variables, such as education, experience, motivation, familiarity with the environment, and above all, stress. Etzioni [103], [104] has proposed a mixed scanning model of decisionmaking that forms the basis for some current research in information systems for planning and decision support [75], [76], [245], [399]. There are a number of contemporary efforts and approaches that support the design of systemic aids that will be more responsive to decisionmaker requirements. Especially important in this regard are the efforts of Einhorn, Kleinmuntz, and Kleinmuntz [95]; Hogarth and Makridakis [161]; Huber [168], Jungerman [181], Kleinmuntz and Kleinmuntz [204], Lad [208], Libby [213], Montgomery and Svenson [256], Payne [271]-[275], Rouse [299], Svenson [364], [365], Thorngate [372], Toda [373]-[375], Tversky and Sattagh [384], Tweney *et al.* [387], Vlek [392], [393]; Wallsten [395], [396]. Efforts which concern the integration of descriptive and prescriptive components of decisionmaking [142], [307], [322], [323]; efforts which concern determination of cognitive choice models in realistic settings [22], [88], [157], [158], [314], [325]; efforts which involve formulation and structuring of decision situations [3], [247], [248], [255], [265], [286], [287], [300], [301], [302], [353], [397]; and efforts which involve the cognitive effort involved in decisionmaking [180], [328], may offer much promise as well.

V. Contingency Task Structure Models

The designer of information systems for planning and decision support must be concerned both with normative models of decision and choice processes and with descriptive models of how people perform, and can perform, in given situations. Thus our discussions of information processing and decision or evaluation rule selection in the previous two sections take on particular meaning in that they comment on the wide variety of possible behaviors. We will be especially concerned, in this section, with describing cognitive processes as they are influenced by the contingency structural elements of task, environment and the human problem solver's experience with these. There have been a limited number of efforts to describe these such as those by Allais and Hagen [3], Beach and Mitchell [32], Borgida and Nisbett [42], Broadbent [49], Bunn [53], Carrol [56], Dreyfus and Dreyfus [82], Einhorn [96], [97], Einhorn and Hogarth [98], Hansanyi [147], Hauser [149], Howell and Fleishman [165], Huber [168], Janis and Mann [176], [177], Jungerman [182], Klein [202], [203], Kleinmuntz and Kleinmuntz [204], Kunreuther and Schoemaker [207], MacKinnon and Wearing [233], Montgomery and Svenson [256], Payne [272], [275], Sage [308], Simon [338], [340], [341], [343], Soelberg [353], and Wallsten [395], [396] This is an area in which additional research could pay major dividends in ultimately increasing the effectiveness of information systems in coping with the contingency task structure variables in planning and decision support.

The contingency task structure model we first describe is related to Piaget's theory of intellectual development [43], [126], [131], [205], [262], [362]. After a description of this model [308] we indicate implications for information system design and the relationship of this model to models that have been proposed by others.

Insights into the nature of cognitive development and insights into a conceptual model of cognitive activity is contained in the works of Piaget, the founder of "genetic epistemology." According to Piaget, there are four stages of intellectual development:

1) sensory motor
2) preoperational
3) concrete operational
4) formal operational.

The last two of these are of particular importance to our efforts here. In the writings of Piaget, intellectual development is seen as a function of four variables:

1) maturation
2) experience
3) education
4) self regulation—a process of mental struggle with discomforting information until identification of a satisfactory mental construction allows intellectual growth or learning.

In Piaget's model of intellectual development concrete operational thinkers can deal logically with empirical data, manipulate symbols, and organize facts towards the solution of well structured and personally familiar problems. Formal operational thinkers can cope in this fashion also. A major difference, however, is that those concrete thinkers who are not also capable of formal thought lack the capacity to reason hypothetically and to consider the effect of different variables or possibilities outside of personal experience. Concrete operational thinkers, for instance, will often have difficulty in responding "true" or "false" to the statement, "six is not equal to three plus four." As another example: "A card has a number on one side and a letter on the other; test the hypothesis that a card with a vowel on one side will have an even number on the other side." Concrete operational thinkers will have difficulty selecting cards for bottom side examination if the top sides of four cards are a, b, 2, 3. However, failure to pick the cards with "a" and 3 on top may not indicate inability as a formal operational thinker but, rather, failure to properly diagnose the task and determine the need for formal operational thought.

We wish to develop a model of higher order cognitive processing that describes the mature adult decisionmaker. Such a decisionmaker will typically be capable of both formal and concrete operational thought. As we will argue, selection of a formal or concrete cognitive process will depend upon the decisionmaker's diagnosis of need with respect to a particular task. That need will

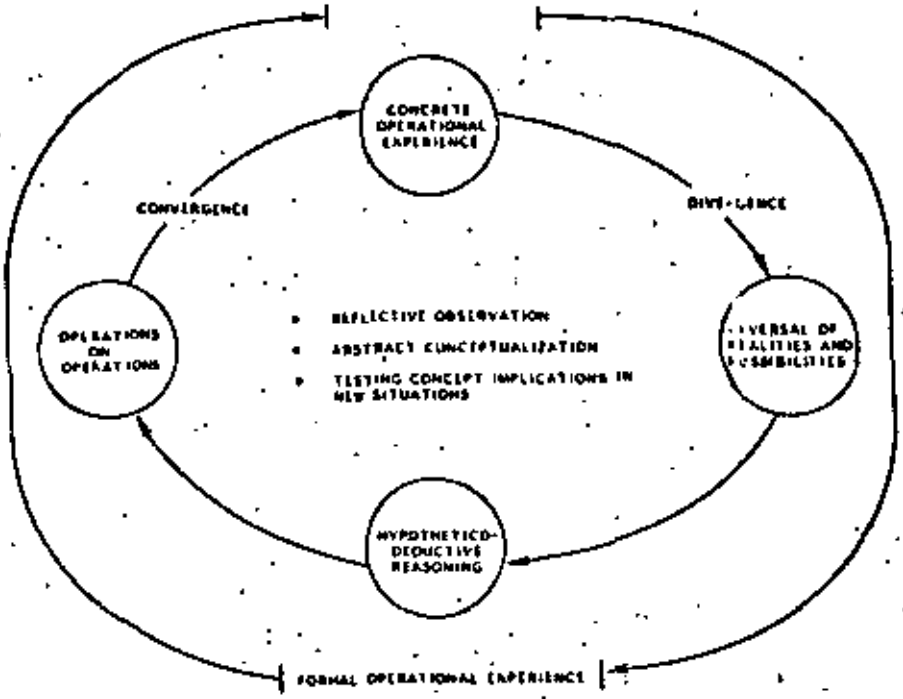


Fig. 6 Learning through formal operational experiences.

depend upon a decisionmaker's maturity, experience, and education with respect to a particular problem. Each of these influence cognitive strain or stress, a subject that will be discussed later in this section. Ordinarily, a decisionmaker will prefer a concrete operational thought process and will make use of a formal operational thought process only when concrete operational thought is perceived inappropriate. In general, a concrete operational thought process involves less stress and may well involve repetitive and previously learned behavioral patterns. Familiarity and experience, with the issue at hand or with issues perceived to be similar or analogous, play a vital role in concrete operational thought. In novel situations, which are unstructured and where new learning is required, formal operational thought is typically more appropriate than concrete operational thought.

We see, in the foregoing discussion, the dominant role of the contingency task structure in guiding problem solving efforts. In concrete operational thought, people use concepts which

1) are drawn directly from their personal experiences,
2) involve elementary classification and generalization concerning tangible and familiar objects,
3) involve direct cause and effect relationships, typically in simple two-variable situations;
4) can be taught or understood by analogy, algorithms, affect, standard operating policy, or recipe; and which
5) are "closed" in the sense of not demanding exploration of possibilities outside the known environment of the person and stated data.

In formal operational thought, people use concepts which may

1) be imagined, hypothetical, based on alternative scenarios, and/or which may be contrary to fact;
2) be "open ended" in the sense of requiring speculation about unstated possibilities,
3) require deductive reasoning using unverified and perhaps flawed hypotheses,
4) require definition by means of other concepts or abstractions that may have little or no obvious correlation with contemporary reality, and which may
5) require the identification and structuring of intermediate concepts not initially specified.

Formal operational thought involves three principal stages:

1) reversal of realities and possibilities
2) hypothetico-deductive reasoning
3) operations on operations

as shown in Fig. 6. These are accomplished through reflective observation, abstract conceptualization, and the testing of the resulting concept implications in new situations. It is in this way that the divergence produced by discomforting new experiences allows the learning of new developments and concepts to be "stored" in memory as part of one's concrete operational experiences.

A number of the cognitive style investigations discussed in Section II have concluded that "abstract" decisionmakers are more information oriented and would typically process much information in complex decision environments. "Concrete" decisionmakers, on the other hand, could be expected to reach an information overloaded state at lower levels of environmental complexity; hence they would tend to process less information than would the abstract decisionmaker. Some models of cognitive style are based on the assumption that "concrete" decisionmakers need more information to arrive at a decision than do "abstract" decisionmakers, suggesting that "concrete" decisionmakers do not give existing information its full worth and more are prone to fits of skepticism than "abstract" decisionmakers. At first glance, cognitive style models such as the one suggested here appear to run in parallel to Piaget's concepts of concrete operational and formal operational thought. But there are very important and very significant differences. These are explicable through the contingency task structure and concept of task, environment, and decisionmaker, a concept that appears, with some notable exceptions, missing in much of the existing cognitive style research cited in Section II.

The concrete operational thinker does not necessarily have limited abilities to process or integrate information, and the "formal" operational thinker is not necessarily capable of "abstract" thought in the specific contingency situation at hand. The formal thinker is neither necessarily able to process information which encompasses more complexity nor better able to cope with uncertainty and disjointedness in the decision environment than is one who uses concrete operational thought in a given decision

situation. Our contingency task structural model for the mature, perhaps expert, adult decisionmaker is one in which the decisionmaker may use formal or concrete operational thought based primarily on diagnosis of the contingency structure of the decision situation, and the stress that is perceived to be associated with the decision situation. This election of a formal or concrete operational mode of thought may be appropriate or inappropriate.

Systemic process design must be responsive to the observation that there are two fundamentally different thought or cognition processes. These are often associated with different halves of the brain [38], [67], [120], [246–248], [254], [410]. One type of thought process is described by the adjectives verbal, logical, sequenced, thinking, and analytical; whereas the second is described as nonverbal, intuitive, wholistic, feeling, and heuristic. The verbal process is typically viewed as superior in engineering and natural science. But this viewpoint on the nature of thought appears wrong and should be discouraged as positively harmful. For the two processes are complementary and compatible. They are not competitive and incompatible in any meaningful way. One thought process may be deficient, in fact, if it is not supported by the other. The nonverbal supports the verbal by suggesting ideas, alternatives, etc. The verbal supports the nonverbal by expressing, structuring, analyzing, and validating the creative ideas that occur in the nonverbal process. An appropriate planning and decision support process must provide for verbal and nonverbal support. An appropriate planning and decision support process must be tolerant and supportive of a decisionmaker's cognitive (thought) processes. These will typically vary across individuals and within the same individual as a function of the environment, the individual's previous experience with the environment, and those associated factors which introduce varying amounts of stress. Thus a contingency task structural view of individuals and organizations in decision situations is needed as contrasted with a stereotypical view in which individuals are assumed to process fixed, static, and unchanging cognitive characteristics which are uninfluenced by environmental considerations.

This view will encourage us not only to consider the evolution of future events over time as inherently probabilistic, but also to consider value change over time. It is especially important that we consider values as containing noncommensurate, ambiguous, and uncertain components, rather than as being absolute, consistent, precise, and exogenous with respect to choice [236], [238].

Typically we learn from experience and adopt various decision rules in the form of cognitive heuristics based upon this experience. The strength of belief that we have in the usefulness of heuristics is often based on reinforcement through feedback. Einhorn [96], [97] has described several supporting illustrations of this. As we have indicated in Section IV. The use of various types of lexicographic semiorders often lead to intransitive choices which are often not recognized as intransitive. We often define issues by content rather than structure and convince ourselves to like what we get from a decision. As a consequence we find it hard to separate decisions from outcomes in retrospective evaluations of our judgments. Much of this is probably due to changing our attitudes and our perceptions in a very selective way without being aware of the change and to changing our forecasts, retrospectively, to correspond to events that have occurred without recognizing this change [117]–[119]. Thus we adopt a hindsight or "knew it all along" bias influenced by a variety of highly selective perceptions of reality.

We are most likely to have coherent value preferences and are able to develop and utilize appropriate evaluation heuristics in well-structured situations with which we are familiar. Learning by trial and error and development of judgment based on either reasoning by analogy, standard operating procedures, or organizational rules typically results from these "concrete operational" situations and experiences. Long standing use of these "rules" results in purely affective judgment and decision responses. In a familiar and simple world, a "concrete operational" world, these judgment guides and judgment heuristics might well be, and in fact often are, quite acceptable. In a changing and uncertain environment, an environment that is different from the one with which we are familiar, we may well err considerably by using these concrete operational world appropriate judgment heuristics. If we do not have a developed set of coherent values relative to a changing environment, we may respond affectively with the first alternative option that comes to mind. We may well adopt postdecision behavior such as to support and maintain a chosen response and employ cognitive biases and cognitive heuristics to justify this potentially ill-chosen response. This results in an affective response appropriate for a "concrete operational" situation when an analytical response, appropriate for a "formal operational" situation, is needed. In the Janis and Mann [177] terminology we adopt a coping pattern based on unconflicted adherence or unconflicted change whereas vigilance is called for.

A serious problem in practice is that we get used to very simple heuristics that are appropriate for "concrete operational" situations in a familiar world and we continue to use them in "formal operational" situations in an unfamiliar world in which they may be very inappropriate. A typical heuristic is incrementalism: "Go ahead and crowd one more beast into the commons." Such a heuristic may be appropriate in the familiar situation our forbears encountered in a new unexplored continent. But the "social traps" produced by such judgmental heuristics in a now crowded environment may be inappropriate. There are numerous contemporary issues to support this assertion.

Styles or modes of information processing, which includes information acquisition and information analysis, are of much importance in the design of information systems for interpretation of the impacts of proposed policy. Information acquisition refers to the perceptual process by which the mind organizes the verbal and visual stimuli that it encounters. As indicated in Section II, McKenney and Keen [242] discuss two modes of information acquisition, a preceptive mode and a receptive mode. We essentially utilize these modes for our model of information acquisition and analysis.

a) In preceptive acquisition and analysis, individuals bring existing experiential concepts and precepts to bear to filter data. They focus on structural relations between items and look for deviations from their expectations. They then use formal precepts as cues for acquisition, analysis, and associated structuring of data.
b) In receptive acquisition and analysis, individuals focus on contextual detail rather than presumed structural relationships. They infer structure and impacts from direct and detailed examination of information, generally including potentially disconfirming information, rather than from fitting it to their precepts.

There is nothing inherently good or bad in either mode of information acquisition, analysis, and associated structuring. The same individual may use different modes as a function of contingency task structure. Most people will have preferences for one mode or the other in a particular situation, depending upon their diagnosis of the contingency task structure and perceived needs to accomplish effective information interpretation and associated decisionmaking. It is our hypothesis that cognitive biases often arise, or are initiated, by use of a situationally incorrect mode of information acquisition and structuring. To use preceptive acquisition when receptive acquisition is more appropriate would appear to invite one or more of the many biases associated with selective perception. To use receptive acquisition when preceptive acquisition is appropriate would appear to introduce much stress associated with the low likelihood of being able to resolve an issue in the time available.

Information evaluation and interpretation refers to the decision rule portion of the problem solution. We advocate a model

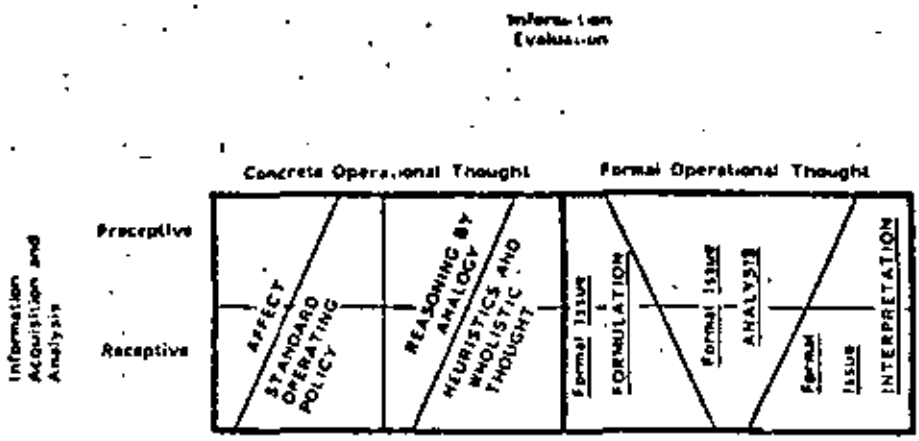


Fig. 7. Conceptualization of problem solving styles.

based on the use of the Piaget theory of concrete and formal operational thinking as a useful precept for information evaluation and interpretation. These thought process models may be summarized as follows.

a) In concrete operational thought, individuals approach problems either through intuitive affect, analogic reasoning, or through following a standard operating policy or organizational processes, or some related process.
b) In formal operational thought, individuals approach problems through structuring in terms of imbedding realities into possibility scenarios, hypothetico-deductive reasoning, and interpretation in terms of operations on operations.

Fig. 7 presents our conceptualization of information acquisition, analysis, and interpretation or problem solving styles. This figure does not illustrate, however, the fundamentally dynamic nature of this process model. Fig. 6 has presented some of the dynamic learning experiences which link the concrete operational and formal operational thought processes. Again we argue that no style is inherently appropriate or inappropriate. Appropriateness of a particular style, as has been mentioned before, is very much task, environment, and experience dependent. That most decisionmakers function as concrete operational thinkers is doubtlessly correct. A principal task of a well-designed information system is to assist in aiding the decisionmaker to detect the appropriate style for a given task, environment, and decisionmaker experience level. Another task is to enhance transfer of formal operational experiences to concrete operational experiences, such as through conceptualization and evolution of appropriate heuristics, wholistic thought, analogous reasoning guides, standard operating procedures, other forms of affective thought, and perhaps even precognitive responses. We posit that both types of information acquisition and analysis may occur with either concrete or formal thought although the appropriate balance of receptive and preceptive acquisition and analysis will vary from situation to situation.

Our discussions have indicated the strong environmental dependence of the formulation, analysis, and interpretation steps necessary for problem resolution. These steps are necessary steps in the resolution of any issue using systemic means, regardless of the "style" adopted for problem solution. Environments, organizations, and technologies are three dominant concerns of systems engineering in general and for the design of systems for planning and decision support in particular. It is the interaction of the environment with an organization and a technology that results in a management technology. Systems management is the term we use to denote the interaction of human judgment with methodological concerns [305]-[308]. Systems management denotes, therefore, concerns at the cognitive process level that involve the contingency task structure and its role in influencing the selection of performance objectives and decision rules for evaluation of options associated with issue resolution. There are many influences which act on the contingency task structure. Fig. 2 indicates, conceptually, how the contingency task structure, and the environment which influences it, acts to specify and direct problem solving efforts through selection of performance objectives and associated information processing and decision rule paradigms.

It is our belief that the dynamic cognitive style models of Figs. 6 and 7 can be used as guides to illustrate both those modes of information acquisition and information evaluation that should be used and that will be used on a given issue. We stress that the particular cognitive style most appropriate for a given issue will depend upon the decisionmakers familiarity with a given issue, the issue itself, and the environment into which the issue is imbedded. Thus a receptive or preceptive information acquisition style will be appropriate in a formal operational setting if the issue at hand is an unfamiliar and unstructured one. The appropriate balance between preceptive and receptive information acquisition will be dependent upon the type of issue and the experience or familiarity the decisionmaker has with possible information sources and their likely reliability. It will, of course, also be influenced by the "personal" style of the decisionmaker and the type, if any, of interaction with the systems analyst as well as upon other characteristics of the decision situation. We accept the view that systems methodologies, especially as implemented through use of human judgment to form a systemic process, are highly value dependent. Different systems methodologies allow one to define issues in different ways and are responsive in differing amounts to value concerns, such as equity. Some methodologies explicitly encourage for example, detection of the use of deficient heuristics and encourage correction. The "transparency" and communicability of a decision process, for example, is very much a function of the methodologies used in process aiding for the formulation of issues, the analysis of alternatives, and associated interpretation efforts. This value dependence of systems methodologies is, therefore, an important aspect of information system design and is related to performance objectives for the task at hand.

There have been a number of studies which focus upon the critical importance of task description and the decisionmaker's interaction with the task through the environment. Dawes [70], [71] stresses the critical interaction among the mind and the task, and integrated models of the mind and the task requirements. He discusses the "even numbered-vowel" experiment described earlier in this section as does Anderson [7]. Anderson indicates that the failure, and a majority of educated adults do fail, to correctly resolve this task is due to difficulties in applying the *modus tollens* concept of conditional deductive reasoning, a concept which requires thinking about what is not the case. Anderson also

discusses a slight variation of this task, which is generally the same, and in which almost all subjects performed correctly. The task involved looking at four pictures of ordinary letter envelopes with the possibility of a stamp on them and picking the letters which should be turned over to test the hypothesis; if a letter is sealed, it has a 18¢ stamp on it. The critical difference between the two tasks is the fact that most people have experiences similar to the second task. It is relatively familiar compared to the first task, concerning which people do not have significant experience.

We should be rather cautious however in the apparently reasonable inference that we learn *correctly* from experience. A number of important studies by Brehmer [46], [47] have shown that by no means do people always improve their judgment and decisionmaking ability on the basis of increased experience. Biases, such as the tendency to use confirming evidence to the neglect of disconfirming evidence, are the key culprits. Brehmer [47] indicates how these biases can be understood in terms of available information. He concludes that truth is not manifest. It needs to be inferred in order to extract from experience information components that will truly lead to better judgments and decisions. The recent definitive discussion of judgment and choice processes by Einhorn and Hogarth [98] emphasizes the importance and the interdependence of attention, memory, cognitive representations, learning, conflict, and feedback. It provides much valuable perspective concerning the importance of these topics for judgment and decisionmaking.

Carroll [56] is much concerned also with understanding decision behavior, especially through the process tracing techniques that have been emphasized by Payne [272]–[275]. Carroll proposes that the decisionmaker might better be portrayed as possessing a rich store of knowledge organized around a variety of evoked schemas, those complex units of organized knowledge which guide the acquisition and use of case information, rather than exclusively considering the decisionmaker as exhaustively following the prescriptions of normative models. Many of the chapters in the recently edited works of Estes [100], Hamilton [137], Howell [167], Howell and Fleishman [165], Schweder [332], and Wallsten [396] discuss issues related to cognitive factors in judgment processes, including task descriptions for scripts, those stereotypical sequences of actions and event schemas which often are of much use in explaining judgment.

Studies of information support for U.S. Air Force command and communication systems accomplished by Klein [202], [203] express a number of concerns regarding artificial intelligence and information processing approaches for decision aiding. These reservations concern potential inabilities of humans to disaggregate situations into components and to analyze these discrete components. He indicates that the proficient performance of experts may well be based more on reasoning by analogy than by representations in terms of step by step descriptions capable of (discrete) digital computer processing. Further, expert proficient performers may not follow explicit conscious rules. Requiring them to do so may reduce performance quality, and they will be unable to accurately describe the rules that they do follow. Klein views expertise as arising from perceptual abilities including recognitional capacity in terms of analogous situations, sensitivity to environmental context in the sense of appreciation of the significance of subtle variations, and sensitivity to intentional context by viewing the relevance and importance of task components as a whole by anticipating what has to occur to achieve a goal rather than just what will occur at the next time instant or step. He presents a comparison-guided model of proficient decisionmaking. In this model [203]

1) a current decision situation is perceived in terms of objectives;
2) the decisionmakers' experience allows recognition of a comparison situation;
3) similarities and differences between the comparison situation and the current situation are noted;
4) this application suggests options, including evaluation of options and selection of a preferred option based on what worked in the comparison option; and
5) the way the objectives and the decision are perceived, possible further adjustments of options, generation of new options, and combination of options, follow from this.

Klein strongly encourages development of decision aids to support the recognitional capacity of the expert; aids that will assist the expert in recognizing new situations in terms of analogous comparison cases and in using these to define options or alternatives. The adjuvant would also keep track of options, assist in generation of new ones, and perform computations to assess the impacts of various options. It certainly appears that this is a needed and necessary role for information systems adjuvants for planning and decision support. But it must be remembered that not all users of such a system will be proficient and expert in all of the tasks they are to perform. We suggest the need also for provisions for formal operational thought type processes for those contingency task situations that have not been sufficiently cognized such that appropriate use of concrete operational thought necessarily leads to efficient and effective performance.

Dreyfus and Dreyfus [82] also argue that experienced and expert human decisionmakers solve new problems primarily by seeing similarities to previously experienced situations in them. They argue strongly that since similarity based processes actually used by experienced and expert humans lead to better performance than formal approaches practiced by beginners, decisionmaking based on proven expertise should not be replaced by formal models. They pose a model which contains five developmental stages through which a person passes in acquiring a skill such as to become a proficient expert. Their basic tenet is that people demand less and less on abstract principles and more and more on concrete experience as they become proficient. Their five stages and suggested instruction at each stage are as follows.

1) *Novice*—Decompose the task environment into context-free nonsituational features which the beginner can recognize without experience. Give the beginner *rules* for determining action and provide monitoring and feedback to improve rule following.

2) *Competence*—Encourage aspect recognition not by calling attention to recurrent sets of features, but rather by singling out perspicuous examples. Encourage recognition of dangerous aspects and knowledge of *guidelines* to correct these conditions. Equal importance weights are typically associated with aspects at this stage.

3) *Proficiency*—This comes with increased practice that exposes one to a variety of whole situations. Aspects appear more or less important depending upon relevance to goal achievement. Contextual identification is now possible and memorized principles, called *maxims*, are used to determine action.

4) *Expertise*—The repertoire of experienced situations is now vast, such that the occurrence of a specific situation triggers an *intuitively* appropriate action.

5) *Mastery*—The expert is *absorbed* and no longer needs to devote constant attention to performance. There is no need for self monitoring of performance and energy is devoted only to identifying the appropriate perspectives and appropriate alternative actions.

Dreyfus and Dreyfus associate the development of these five skill categories with successive transformation of four mental functions. Fig. 8 [82] indicates how these transformations occur with increased stages of proficiency. While developed primarily for training this model contains much of importance with respect to information system design to support planning and decisionmaking as well. A key issue in this table would appear to be the development of concrete situational experience which first occurs when a person is able to recognize aspects. There seems to exist some complimentarity between our model of the cognitive judgment and decision process and that of Dreyfus and Dreyfus. The concrete operational thought of experienced decisionmakers

SKILL LEVEL	MENTAL FUNCTION: RECOLLECTION OF SIMILAR FEATURES	RECOGNITION OF ASPECTS	DECISION PARADIGM	AWARENESS OF TASK
NOVICE	NON SITUATIONAL	DECOMPOSED	ANALYTICAL	MONITORING
COMPETENT	SITUATIONAL			
PROFICIENT				
EXPERT		WHOLISTIC	INTUITIVE	ABSORBED
MASTER				

Fig. 8. Dreyfus' judgment and decision process model.

would appear to be much the same as the thought of the expert and the master. Of course in all of these models, "expert" is a relative term, with the environment and the contingency task structure of a specific situation needed to determine whether a decisionmaker is familiar and experienced with it. Some differences in the models are doubtlessly present as well. Some of these depend upon precisely what is meant by "processing information." Our definition is rather broad and certainly not restricted to quantitative processing. Generally information processing, in our view, includes the formulation or acquisition, analysis, and interpretation of data of value for decisionmaking. This can be accomplished holistically, heuristically, or wholistically.

Very important concerns exist, in our view, with respect to possible cognitive bias and value incoherencies in the concrete operational decisionmaking of experts or masters. Questions related to the effects of changing environments upon the judgment and decision quality of masters and novices alike are very important in all of these models. For intuitive experience may not be a good guide for judgments and decisions in uncertain, unfamiliar, and/or rapidly changing environments. But quantitative or qualitative analysis-based efforts may well not be very good either due to changed decision situation and contingency task structural models. In our view it is possible to become a "master," but unfortunately possible to become a master of the art of self-deception as well as of a specific task. The external behavior of the two "masters" may well be the same; situational, wholistic, intuitive, and absorbed. What was an appropriate style for one "master" may well be inappropriate for another.

Behavior in familiar but uncertain environments is of much interest. Studies of failure, situations in which experts and masters fail or misdiagnose their degree of expertise or mastery, could yield exceptionally useful results and would also serve to incorporate and integrate much of the experimental work involving biases, poor heuristics, and value coherences into a more real decision situation. We hypothesize that the dynamic models of decision styles presented in this section will be useful vehicles to these ends.

Judgment and decisionmaking efforts are often characterized by intense emotion, stress, and conflict; especially when there are significant consequences likely to follow from decisions. As the decisionmaker becomes aware of various risks and uncertainties that may be associated with a course of action, this stress becomes all the more acute. Janis and Mann [176], [177] have developed a conflict model of decisionmaking. Conflict here refers to "simultaneous and opposing tendencies within the individual to accept and reject a given course of action." Symptoms of conflicts may be hesitation, feelings of uncertainty, vacillation, and acute emotional stress with an unpleasant feeling of distress being, typically, the most prevalent of all characteristics associated with decisionmaking [49]. The major elements associated with the conflict model are the concept of vigilant information processing, the distinction between hot and cold cognitions, and several coping patterns associated with judgments.

Cold cognitions are those made in a calm detached environmental state. The changes in utility possible due to different decisions are small and easy to determine. Hot cognitions are those associated with vital issues and concerns, and are associated with a high level of stress. Whether a cognition is, or should be, hot or cold is dependent upon the task at hand and the experiential familiarity and expertness of the decisionmaker with respect to the task. The symptoms of stress include feelings of apprehensiveness, a desire to escape from the distressing choice dilemma, and self-blame for having allowed oneself to get into a predicament where one is forced to choose between unsatisfactory alternatives. Janis and Mann [177] state that "psychological stress" is used as a generic term to designate unpleasant emotional states evoked by threatening environmental events or stimuli. They define a "stressful" event as "any change in the environment that typically induces a high degree of unpleasant emotion, such as anxiety, guilt, or shame, and which affects normal patterns of information processing." Janis and Mann describe five functional relationships between psychological stress and decision conflict.

1) The degree of stress generated by decision conflict is a function of those objectives which the decisionmaker expects to remain unsatisfied after implementing a decision.
2) Often a person encounters new threats or opportunities that motivate consideration of a new course of action. The degree of decision stress is a function of the degree of commitment to adhere to the present course of action.
3) When decision conflict is severe because all identified alternatives pose serious risks, failure to identify a better decision than the least objectionable one will lead to defensive avoidance.
4) In severe decision conflict when the decisionmaker anticipates having insufficient time to identify an adequate alternative that will avoid serious losses, the level of stress remains extremely high. The likelihood that the dominant pattern of response will be hypervigilance, or panic, increases.
5) A moderate degree of stress, which results when there is sufficient time to identify acceptable alternatives, in response to a challenging situation, induces a vigilant effort to carefully scrutinize all identified alternative courses of action and to select a good decision.

Based upon these five functional relation propositions, Janis and Mann present five coping patterns which a decisionmaker would use as a function of the level of stress: unconflicted adherence or inertia, unconflicted change to a new course of action, defensive avoidance, hypervigilance or panic, and vigilance. These five coping patterns, in conjunction with the five functional relation propositions of psychological stress, were used

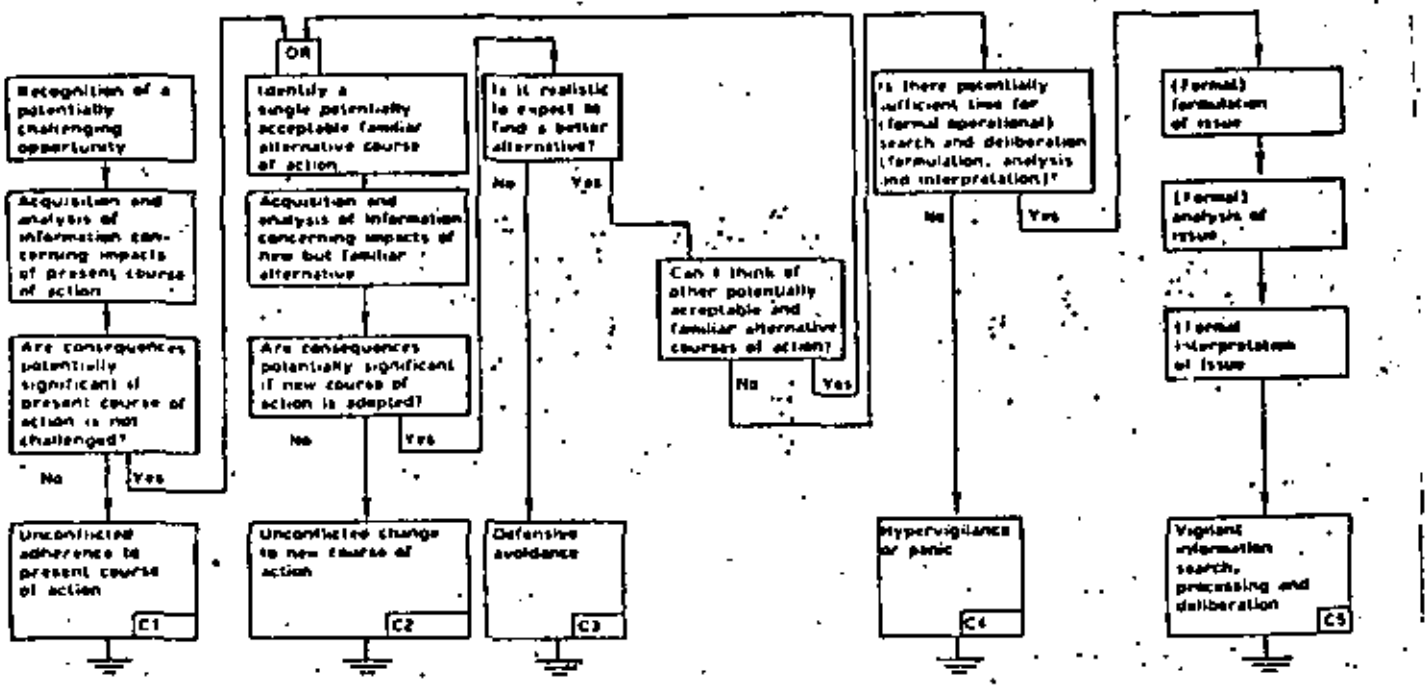


Fig. 9. Interpretation of the Janis and Mann [177] conflict model of decisionmaking.

by Janis and Mann to devise their conflict model of decisionmaking. This model postulates that each pattern of decision stress for coping is associated with a characteristic mode of information processing. It is this mode of information processing which governs the type and amount of information the decisionmaker will prefer. Fig. 9 presents an interpretation of this conflict model of decisionmaking in terms of the systems engineering contingency models discussed in this section. This model points to a number of markedly different tendencies which become dominant under particular conditions of stress. These include open-mindedness, indifference, active evasion of disconfirming information, failure to assimilate new information, and all of the other cognitive information processing biases identified in Section III. Table II summarizes information processing preferences and decision styles generated by this conflict mode. The table depicts the striking complexity entailed by the vigilant information processing pattern in comparison to the other coping patterns. The vigilance pattern is characterized by seven key steps which require somewhat prolonged deliberation. The other four coping patterns require that only a few key steps be addressed. Selection of a coping pattern may be made properly or unwisely, just as selection of a decision style may be proper or improper. The seven steps of vigilant information processing appear quite equivalent to the steps of systems engineering.

Janis and Mann [177] combine the hypotheses they present concerning the four stages of the decisionmaking (which we discuss in Section I), the five functional relation propositions of psychological stress, and the five stress coping patterns. Also they present a decision balance sheet, an adaptation of the moral algebra of Benjamin Franklin [177], on which to construct a profile of the identified options together with various cost and benefit attributes of possible decision outcomes. They have shown that decision regret reduction and increased adherence to the adopted decision results from use of this balance sheet. Strategies for challenging outworn decisions and improving decisions quality are also developed in this seminal work.

It would be of considerable interest to indicate the typical interactions between this model of Janis and Mann, which would be an expanded version of Fig. , and the other three contingency task structure models of decision style that we have discussed in this section. We believe each of these models to be appropriate and to portray different relevant features of task evaluation, information processing preference, and decision rule selection in terms of contingency elements associated with the environment and the decisionmaker's prior experiences.

There are, of course, other models of the planning and decisionmaking process. Within the field of artificial intelligence there exists a growing important body of literature concerning models of cognitive processes in planning and decisionmaking [150], [398]. The work of the Hayes–Roths [150] is definitive in this regard. It presents a model of the independent actions of a large number of cognitive specialists who make tentative decisions for incorporation into a tentative plan. Different specialists influence different portions of a plan and reguard decisions concerning the plan on a common data structure called a "blackboard." This blackboard allows specialists to retrieve prior plans and decisions and to combine earlier decisions with present decisions, thereby potentially generating new decisions. A process description of how knowledge generated during issue resolution is structured, stored, and used is available.

The basis assumptions underlying the planning model are: that decisions occur at several different levels of abstraction, that present decisions will constrain subsequent decisions, and that people can adopt alternative and appropriate strategies for planning. This important research suggests areas in which present and prospective planners need training by potentially allowing determination of differences in the information processing and judgment strategies of people as a function of task and environment.

VI. Decisionmaking Frameworks and Organizational Settings

We have already discussed such topics as decisionmaking rules, cognitive styles, information processing and contingency task structural models. Each of these represents a necessary component in the description of components of the decisionmaking process. While these components are all necessary for understanding of the decision process, they are not sufficient. In particular the nature of the decisionmaking process is very much influenced by the topics to be discussed in this section: various types of reasoning, the degree of approximation to various conceptual models of decisionmaking, the degree of centralization of the decision process, and the effects of these factors upon infor-

mation acquisition. All of these factors are typically related and all are part of the contingency task structure.

Characterizations of Rationality

Diesing [77] is among several writers such as Steinbruner [359], who have defined several forms or types of rationality. Diesing defines five forms of rationality.

1) *Technical Rationality*—This results from efficient achievement of a single goal. A technically rational organization is one in which all of the activities of the organization are efficiently organized to achieve the goal of the organization. Technological progress requires an increase in the efficiency of the productive process and the existence of social conditions that make this increased efficiency possible. Diesing notes that a technological innovation that deals only with more efficient means to a single end will often have rather limited influence if the impacts of the technology and resulting attributes are morally and psychologically isolated from one another.

2) *Economic Rationality*—This results from maximum achievement of a plurality of goals. There are four characteristics needed for existence of an economy. Two of these relate to allocation: plurality of alternative ends, common means to the ends, scarcity of resources; and availability of a value system and associated measurements. Two characteristics relate to exchange: plurality of economic units, and a different prioritization of values among these units. Diesing claims that maximum goal achievement or economic rationality is possible if

a) the ends (goals) of the economic units are comparable and measurable on a single scale;
b) there are no limits on the assignability and use of the means;
c) economic units are integrated enough to engage in rational allocation and exchange; and
d) information about the supply-demand relationships for the various units is available and known to all.

Consequently economizing includes both evaluation and selection of various ends and means. Clearly it is desirable that conditions a)-d) hold but there exist many approaches to maximization under constraints that may be used to yield optimum resource allocation under constraints. Economic progress is equivalent to an increase in productivity per labor hour and, consequently, increased productivity can only result from economic and technical change. Economic progress will typically spread rapidly throughout a culture because it allows more and more ends to become both alternatives to each other and means to other ends as well. Generally the rational actor model we have discussed before is equivalent to Diesing's model of economic rationality.

3) *Social Rationality*—A social system is an organization of cultural roles such as expectations, obligations, and ideals. A social system is said to be integrated when the various associated activities fit well, support, confirm, enrich, and reinforce one another. Social integration is more than mechanical efficiency and consistency due to the mutual support, enrichment, confirmation, and reinforcement requirement. This integration makes action possible by

a) channeling emotional energy and preventing it from being diffused and lost;
b) eliminating conflict which could block action;
c) providing those supporting factors which strengthen action and which allow action to be carried through to completion; and
d) making actions more meaningful by allowing them to be related to past and future actions.

An integrated social system is a rational social system that enhances the meaningful and successful completion of actions. Successfully completed actions are not necessarily either efficient or effective as integration promotes survivability of the system and not necessarily the people within it. In extreme cases of inefficiency or ineffectiveness, people may leave the system and establish another one. Four characteristics of a *rational* social organization, as described by Diesing, are

a) internally consistent roles that can be carried out by the society without great strain;
b) harmonious roles that fit together without conflict among roles;
c) smoothly evolving roles such that there exists continuity and stability with no sharp impulsive changes in roles over time; and
d) roles compatible with the nonsocial (i.e., geographic, technoeconomic, temporal, and physiological) environment.

As it develops and becomes more integrated, a social system develops a value system that reinforces, through feedback, the structure of, and roles within, the social system. Well-integrated socially-rational systems typically resist change and avoid risk in our interpretation of Diesing. One might argue, of course, that a well-integrated social system *should* be adaptive to change and that failure to do so will subject it to a greater long term risk than if it were organically adaptive to change. This is, perhaps, the difference between a descriptive view and a normative view of a well-integrated social system.

4) *Legal Rationality*—A legally rational system is a system of rules which are complex, consistent, precise, and detailed enough to be capable of unambiguous application. Some of these rules may apply impartially to all people, while others may apply differently to different classes of people. A "legally rational" system is rational because, and if, it is effective in preventing disputes. It does this by providing a framework which defines and supports performance of economic and social rules. This framework also provides a procedure for settlement of those disputes which occur.

5) *Political Rationality*—This is the rationality of decisionmaking structures. A decisionmaking structure is composed of a set of discussion relationships and a set of beliefs and values that are imbedded into a set of recognized roles. These roles have been assigned to individuals such as to enable actions within the context of previous actions and commitments. Politically rational decision structures are based upon three guiding imperatives, according to Diesing:

a) maintenance of independence of the group despite all pressures for dependence;
b) actions to structure the political group such that pressures are balanced and moderate; and
c) preparation for future pressures which act to increase the stability and political rationality of the decision structure by providing unification and broadening of participation.

These forms of rationality are certainly related. Technical rationality is necessary for, and a part of, economic rationality. The primary characteristic which follows from rational economic behavior is a detachment or neutrality of intrinsically valueless commodities. These are useful only as means to ends such that scalar optimization may be used to select the commodity bundle of alternate means. Particularism and loyalty are the primary characteristics of social rationality such that obligations evolve from particular social relations with individuals and groups rather than general, universalist-detached relations which are applicable to all. Ascription, in which actions towards people evolve from particular relations rather than as a response to achievement, is another characteristic of social rationality. Thus we see that the characteristics of economic rationality *may* contrast sharply with those of social rationality. But this, we believe, is not necessarily the case. For, as Diesing indicates, neither form of rationality can exist without some form of the other. Economic rationality theories are based on the assumption that social integration is a

reality: such that there exist communication and valuation capabilities and no goal conflicts or factionalism. In a similar way social rationality assumes that societies' economic resource allocation problems are solved.

Social and political rationality are related in the sense that both are primarily concerned with internal structural concerns involving process and procedure; that is, the structure of interpersonal relations, or the accumulation of power, or the direction of pressure. Economic and legal rationality are primarily concerned with the substantive behavior as contrasted with procedural and internal structural concerns. We have argued strongly in previous sections that substantive and procedural rationality [206], [336] are each necessary considerations in information system design.

Decision Frameworks

We have presented a detailed synopsis of the perceptive work of Diesing [77] concerned with five different forms of rationality. Additional forms of rationality [50], perhaps based upon the ten interacting societal sectors noted [304], [307] could doubtlessly be developed. It would be of interest to determine the extent to which these additional forms of rationality would be subsets of and independent of the five forms of Diesing.

March [236] discusses bounded rationality, limited rationality, contextual rationality, game rationality, and process rationality. A study of relations of these forms of rationality both to the rationality forms of Diesing and decision frameworks, which we will now discuss, could lead to useful insights and more relevant systemic process designs.

The organizational science literature contains much discussion relative to the development of conceptual models for decision-making based upon various rationality conceptualizations. Among these are the (economic) rational actor model, the satisficing or bounded rationality model, the bureaucratic politics, incremental, or "muddling through" model, the organizational processes model, and the garbage can model. These are related to the five types of rationality described by Diesing in relatively obvious ways that follow directly from a description of these decision frameworks.

1) *The Rational Actor Model*: The decisionmaker becomes aware of a problem, studies it, carefully weighs alternative means to a solution and makes a choice or decision based on an objective set of values. This is comparable to technical and economic rationality as described by Diesing. At first glance the rational actor model appears to contain much of value and to be especially well matched to the detached neutrality, calculative orientation, and avoidance of favoritism associated with the achievement-oriented entrepreneurial western society. In rational planning or decisionmaking

a) the decisionmaker is confronted with an issue that can be meaningfully isolated from other issues;
b) objectives are identified, structured, and weighted according to their importance in achieving need satisfaction on various aspects;
c) possible activities to resolve needs are identified;
d) the impacts of action alternatives are determined;
e) the utility of each alternative is evaluated in terms of its impacts upon needs; and
f) the utilities of all alternatives are compared and the policy with the highest utility is selected for action implementation.

These are essentially equivalent to the vigilant information processing steps of Janis and Mann [177].

Unfortunately, there are several substantive requirements for successful *complete* rational decisionmaking that will not generally be met in practice. These include

a) comprehensive identification of *all* needs, constraints, and alterables relevant to planning and decisionmaking is, of course, not possible;
b) determination and clarification of *all* relevant objectives is, of course, not possible;
c) determination and minimization of costs and maximization of effectiveness will not necessarily lead to the "best" results because of a) and b);
d) detached neutrality and a calculative orientation rather than arbitrariness, conflict, and coercion are not always possible;
e) a unified process that will cope with interdependent decisions will often be very complex;
f) sufficient time to use the method will often not be available;
g) sufficient information to enable use of the method will often be difficult and expensive to obtain; and
h) sufficient cognitive capacity to use the method will often not exist.

It has long been recognized by systems engineers and management scientists that the attempt to use a normatively optimum process will result in less than optimum results because of these modeling inaccuracies, cognitive limitations, and solution time constraints. Thus the presence of the realities of a)-h) will, because of a combination of resource and intellectual constraints, lead to selection of an alternative that is best only within constraints posed by the model actually used. We may also observe that an economically rational decision would only be appropriate when the decision situation structural model is such that an economically rational process is possible and desirable, and that the intellectual and resource conditions extant make substantive use of the rational actor model feasible.

Simon [337], [339], [340], [343] was perhaps the first to observe that unaided decisionmakers may not be able to make complete substantive, that is "as if," use of the model possible. The concepts of bounded rationality and satisficing represent much more realistic substantive models of actual decision rules and practices. We have described a variety of satisficing heuristic rules in Section IV. Unless very carefully developed and applied, these rules may result in very inferior decisions; decisions which are reinforced through feedback and repetition such as to result in experiences that are, by no means, the best teacher.

Of possibly even greater importance to information system design is the fact that completely economically rational processes may be neither desirable nor possible. Social, political, or legal rationality concerns may well prevail. And one of the other decision frameworks we describe here may well be more appropriate if these concerns are dominant over economic rationality concerns.

2) *The Satisficing or Bounded Rationality Model*: The decisionmaker looks for a course of action that is basically good enough to meet a minimum set of requirements. The goal is to "not shake the system" or "play it safe" by making decisions primarily on the basis of short-term acceptability rather than seeking a long-term optimum.

Simon introduced the concept of satisficing or bounded rationality as an effort to

> replace the global rationality of economic man with a kind of rational behavior that is compatible with the access to information and the computational capabilities that are actually possessed by organisms, including man, in the kinds of environments in which such organisms exist.

He suggested that decisionmakers compensate for their limited abilities by constructing a simplified representation of the problem and then behaving rationally within the constraints imposed by this model. The need for this rests in the fact that many decisionmakers satisfice by finding either optimum solutions in a simplified world or satisfactory solutions in a more realistic world. As Simon says, "neither approach dominates the other [341]".

Satisficing is actually searching for a "good enough" choice. Simon suggested that the threshold for satisfaction, or aspiration level, may change according to the ease or difficulty of search. If

many alternatives can be found the conclusion is reached that the aspiration level is too low and needs to be increased. The converse is true if no satisfactory alternatives can be found. This may lead to a unique solution through iteration.

The principle of bounded rationality and the resulting satisficing model suggests that simple heuristics may well be adequate for complex problem solving situations. Satisficing strategies may be excellent for repetitive problems. They may also lead to premature choices that result in unforeseen disasterous consequences; consequences which could have been foreseen by more careful analysis. The heuristic decision rules described in Section IV are all versions of satisficing strategies. A recent paper by Thorngate [372] provides useful descriptions of ways in which heuristic decision rules may be used and abused. Development of efficient and effective decision heuristics is a contemporary need for the analysis of decision behavior [56], [59], [60], the modeling of organizational and individual decisions [292], [365] as well as for the design of normative systems to aid decisionmaking [316]. We believe that to be effective as well as efficient, heuristics will have to be developed in a very cautious way with due considerations for the many implications of the contingency task structure of a decision situation [326].

3) *The Bureaucratic Politics, Incrementalism, or "Muddling Through" Model:* After problems arise which require a change of policy, policymakers consider only a very narrow range of alternatives differing to a small degree from the existing policy. One alternative is selected and tried with unforseen consequences left to be discovered and treated by subsequent incremental policies. This is the incremental view.

In 1959 Lindblom postulated the approach called incrementalism, or muddling through [218]–[221], to cope with perceived limitations in the economically rational approach. Marginal values of change only are considered—and these for only a few dimensions of value whereas the rational approach calls for exhaustive analysis of each identified alternative along all identified dimensions of value. A number of authors have shown incrementalism to be the typical, common, and currently practiced process of groups in pluralistic societies. Coalitions of special interest groups make cumulative decisions and arrive at a workable compromise through a give and take process that Lindblom calls "partisan mutual adjustment." He indicates that ideological and other value differences do not influence marginal decisions as much as major changes and that, in fact, considering marginal values subject to practical constraints will lead to agreement on marginal programs. Further, incrementalism can result in agreement on decisions and plans even by those who are in fundamental disagreement on values. However incrementalism appears based on keeping the masses marginally content and thus may not be able to do much to help the greatly underprivileged and unrepresented. It is, of course, a combination of Diesing's social and political rationality. Boulding has compared incrementalism to "staggering through history like a drunk putting one disjointed incremented foot after another." Yet there have been a number of studies, such as Allison's study of the Cuban missile crisis [4], Steinbruner's case studies [359], and others [44], [108], [135], [400] which indicate this to be an often used approach in practice.

It is important to note [218] that Lindblom rejects (economic) comprehensive rationality even as a normative model and indicates that systems analysis will often lead to ill-considered, often accidental incompleteness. He indicates the following inevitable limitations to analysis.

a) It is fallible, never rises to infallibility, and can be poorly informed, superficial, biased, or mendacious.
b) It cannot wholly resolve conflicts of value and interests.
c) Sustained analysis may be too slow and too costly compared with realistic needs.
d) Issue formulation questions call for acts of choice or will and suggests that analysis must allow room for politics.

A perceived more practical model process for decisionmaking than the rational actor model is, therefore, called for. The model is descriptive and is an extreme version of the bounded rationality model. Alternative models have been proposed [317].

The main features of the model proposed by Lindblom are the following.

1) Ends and means are viewed as not distinct. Consequently means-ends analysis is viewed as often inappropriate.
2) Identification of values and goals is not distinct from the analysis of alternative actions. Rather, the two processes are confounded.
3) The test for a good policy is, typically, that various decisionmakers, or analysts, agree on a policy as appropriate without necessarily agreeing that it is the most appropriate means to an end.
4) Analysis is drastically limited, important policy options are neglected, and important outcomes are not considered.
5) By proceeding incrementally and comparing the results of each new policy with the old, decisionmakers reduce or eliminate reliance on theory.
6) There is a greater preoccupation with ills to be remedied rather than positive goals to be sought.

In a very readable recent work concerning "muddling through [221]," Lindblom classified incremental analysis at three levels: simple, disjointed, and strategic. Incremental analysis is, as we have indicated, a good description of political decisionmaking and is sometimes referred to as the political process model.

4) *The Organizational Processes Model:* Plans and decisions are the result of interpretation of standard operating procedures. Improvements are obtained by careful identification of existing standard operating procedures and associated organizational structures and determination of improvements in these.

The organizational process model, originally due to Cyert and March [68], functions by relying on standard operating procedures which constitute the memory or intelligence bank of the organization. Only if the standard operating procedures fail will the organization attempt to develop new standard procedures.

The organizational processes model may be viewed as an extension of the concept of bounded rationality to choicemaking in organizations. It is clearly an application of reasoning and rationality as discovery and application of rules to cases. It may be viewed as a hybrid of economic and legal rationality. It typically involves concrete operational thought as we have indicated in Section V. The main concepts of the behavioral theory of the firm, which is suggested as a descriptive model of actual choicemaking in organizations are as follows.

1) Quasiresolution of conflict: Major problems are disaggregated and each subproblem is attacked locally by a department. An acceptable conflict resolution between the efforts of different departments is reached through sequential attention to departmental goals.
2) Uncertainty avoidance is achieved
   a) by reacting to external feedback,
   b) by emphasizing short-term choices, and
   c) by advocating negotiated futures.
3) Problem search in which
   a) search is stimulated by encountering issues,
   b) a form of "satisficing" is used as a decision rule, and
   c) search in the neighborhood of the status quo only is attempted and only incremental solutions are considered.
4) Organization learning: Organizations adapt on the basis of experience.

The organizational process model may be viewed as suggesting that decisions at time t may be forecasted with almost complete certainty from knowledge of decisions at time $t - T$ where T is the planning or forecasting period. Standard operating proce-

dures or "programs," and education motivation and experience or "programming" of management are the critical determinants of behavior for the organizational process model.

5) The Garbage Can Model: This relatively new model [63] views organizational decisionmaking as resulting from four variables: problems, solutions, choice opportunities, and people. Decisions result from the interaction of solutions looking for problems, problems looking for solutions, decision opportunities, and participants in the problem solving process. The model allows for these variables being selected more or less at random from a garbage can. Doubtlessly this is a realistic descriptive model. It is especially able to cope with ambiguity of intention, understanding, history, and organization. Further, it provides much support for a feedback model of organizational choice in which preferences and cognitions of individuals affect their behavior, behavior of individuals affect organizational choices, organizational choices affect environmental activities and responses, and environmental activities and responses affect the preferences and cognition of individuals [237]. An extensive and definitive discussion of ambiguity and choice in organization, with emphasis upon the garbage can model, is contained in [237].

All five of the models or frameworks for decisionmaking have both desirable and undesirable characteristics. Conclusions may be drawn from these models and the fact that any of them may be relevant in specific circumstances. If we accept the facts that

1) decisionmakers use a variety of methods to select among alternatives for action implementation;
2) these methods are frequently suboptimal; and
3) most decisionmakers desire to enhance their decisionmaking efficiency and effectiveness;

then we must conclude that there is much motivation and need for research and ultimate design and development of planning and decision support systems. But these five models make it very clear that improved planning and decisionmaking efficiency and effectiveness and aids to this end can only be accomplished if we understand human decisionmaking as it is as well as how it might be and allow for incorporation of this understanding in systemic process adjuvants. One of the requirements imposed on these adjuvants will be relevance to the individual and group decisionmaking structure [181], [237], [286], [303], [401]. Another requirement is relevance to the information requirements of the decisionmaker. We discuss both of these in this section of our survey and interpretation.

There have been many studies of group decisionmaking. These include the fundamental theoretical studies of Arrow [17] and others which show that, under a very mild set of realistic axioms, there is no assuredly successful and meaningful way in which ordinal preference functions of individuals may be combined into a preference function for society [17], [196], [279], [302]. Conflicting values [378] are the major culprit preventing this combination. This has a number of implications which suggest much caution in using ordinal preference voting systems and any systemic approach based only on ordinal, possibly wholistic, or heuristic preferences among alternatives. Among other possible debilitating occurrences are agenda dependent results which can, of course, be due to other effects [280]. There have been a number of studies of group decisions and social and organizational interactions such as those by Bacharach [19], Davis [69], Ebert and Mitchell [89], Fishburn, Hogarth, and Klemperer [92], Holloman and Hendrick [152], Janis and Mann [176], [177], Leavitt [211], Mintzberg [248], Penrod and Hastie [276], Schein [312], Shumway, *et al.* [329], Simon [341], Vinokur and Burnstein [390], [391] and in the edited work of Hooker, Leach, and McClennen [163]. Several systematic methods have been proposed for forming and aggregating group opinions as described in the works of Hogarth [155], Huber [169], Hylland and Zeckhauser [173], Rohrbaugh [295], Van de Ven and Delbecq [388]. An excellent survey of voting methods and associated paradoxes is presented by Fishburn [122] and by Plott [279].

Very definitive studies of the interpersonal comparison of utilities have been conducted by Harsanyi [145]–[147]. He argues convincingly that we make interpersonal utility comparisons all the time whenever we make any allocation of resources to those to whom we feel the allocation will do the most good. The prescription against such comparisons is one of two key restrictions which lead to the Arrow impossibility theorem. By using cardinal utilities such that it becomes possible to determine preferences among utility differences (i.e., whether $u(a) - u(b) > u(b) - u(c)$), and interpersonal comparison of utilities, Harsanyi shows that Arrow's impossibility theorem becomes a possibility theorem. This is a major point in that it is generally not possible for a group to express meaningful transitive ordinal preferences for three or more alternatives even though all individuals in the group have individually meaningful transitive ordinal preferences.

Harsanyi is concerned primarily with organizational design [147]: how to design social decisionmaking units so as to maximize attainment of social objectives or value criteria. He shows that rational morality is based on maximization of the average (cardinal) utility level for all individuals in society. The utilitarian criterion is applied first to moral rules and then these moral rules are used to direct individual choices. Thus each utilitarian agent chooses a strategy to maximize social utility under the assumption that all other agents will follow the same strategy. Harsanyi recognizes a potential difficulty [147] with this particular utilitarian theory of morality in that it is open to dangerous political abuses as well as the numerous problems associated with information acquisition and analysis in a large centralized system. He posits a difference between moral rationality and game-theoretic rationality. He argues the unavoidable use of interpersonal cardinal utility comparisons in moral rationality and the inadmissibility of such comparisons in game theory. Much of Harsanyi's efforts concern game situations [146] in which outcomes depend on mutual interactions between morally rational individuals, each attempting to better their own interests. We will not attempt to explore here the very interesting subjects of bargaining, conflict, resolution, and negotiation and the use of systems for planning and decision support to these ends [21], [45], [269].

Harsanyi's concept of utilitarianism has occasionally been criticized for making inadequate provision for equity, or equivalently for social group equality. John Rawls, a philosopher, has presented a theory of justice [291] which involves a difference principle in which decisions are made under uncertainty rather than under risk. This difference principle advocates selection of the alternative choice which is the best for the worst-off member of society and is, therefore, the direct social analog of the maximum principle for the problem of individual decisions under certainty. Rawls uses a "veil of ignorance" concept in which individuals must determine equitable distribution of societies' resources before they know their position in society. His argument is essentially that people will select a resource allocation rule that maximizes the utility of the worst-off member of society. Discussions of some of the potential difficulties associated with Rawls' "social contract" justice theory are presented by Ellsworth and Gauthier [163].

Other useful interpretations of cardinal utility and interpersonal utility comparisons have been made by Keeney and Kirkwood [194] and Keeney [195]. Their axioms allow development of a multiplicative group utility function in contrast to the additive utility function of Harsanyi. It is possible to more directly deal with equity considerations in a multiplicative group utility model than in an additive model. Papers by Bodily, Brock, and Keeney [201] contain insightful discussions concerning group and individual utilities of a multiattribute nature. Ulvila and Snider [201] illustrate use of multiattribute utility models in negotiations.

We are particularly interested, here, in describing decisionmaking efforts in hierarchical organizations [241]. This leads naturally to a study of information processing in organizations

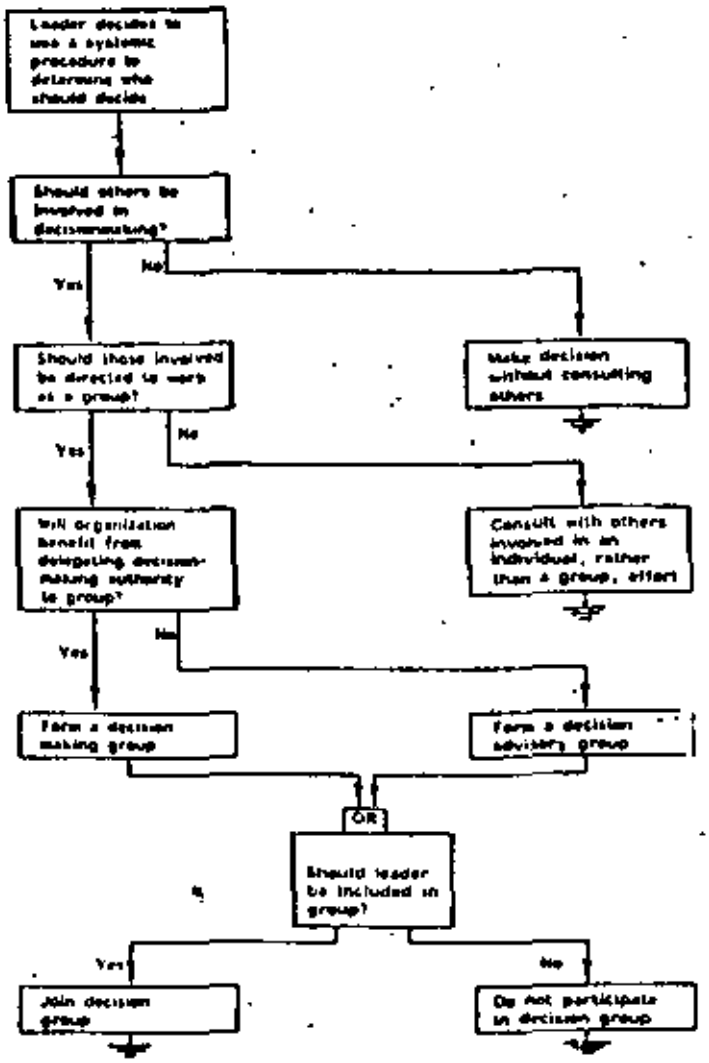


Fig. 10. Delta chart on how to decide who should decide (after [169]).

and a description of how decisionmakers may determine information needs. While there have been a number of studies of group decisionmaking roles and organizational behavior [357], [370], our efforts will be based primarily on those of Vroom and Yetton [394] and Huber [169].

Huber, Vroom, and Yetton have indicated a number of potential advantages and disadvantages to group participation in decisionmaking. Since a group has more information and knowledge potentially available to it than any individual in the group, it *should* be capable of making a better decision than an individual. Group decisions are often more easily implemented than individual decisions since participation will generally increase decision acceptance as well as understanding of the decision. Also group participation increased the skills and information that members may need in making future organizational decisions. On the other hand there are disadvantages to groups. They consume more time in decisionmaking than individuals. The decisions may not fully support higher organizational goals. Group participation may lead to unrealistic anticipations of involvement in future decisions and resentment towards subsequent individual decisions in which they have not participated. Finally, there is no guarantee that the group will converge on a decision alternative.

Huber asks four primary questions the answers to which determine guidelines for selection of a particular form of group decisionmaking. The delta chart of Fig. 10 indicates how the responses to these questions determines an appropriate form of group decisionmaking. There are a number of subsidiary questions concerned with each of the primary questions. For example, we may determine whether or not to involve others by posing questions involving decision quality, understanding, and acceptance, personnel development, and relationships, and time required.

Vroom and Yetton have been much concerned with leadership and decisionmaking [394]. Their primary concern is with effective decision behaviors. They develop a number of clearly articulated normative models of leadership style for individual and group decisions. These should be of use to those attempting to structure normative or prescriptive models of the leadership style portion of decision situations which are capable of operational implementation. We will not illustrate these here since they essentially involve generalizations of Fig. 10. It is the apparent goal of Vroom and Yetton to move beyond generalities such as the leadership style theory *X*-theory *Y* [211]; [394]. They desire to come to grips with and explicitly use leadership behavior and situational variables to enhance organizational effectiveness.

Much of our discussion in this section has concerned the evaluation component of various decisionmaking frameworks and organizational settings. Effective planning and decision support is based not only upon evaluation, but upon information acquisition and processing as well. We have emphasized this in our discussions thus far in terms of individual information processing behavior; but have not yet given explicit consideration to information processing behavior in organizations.

Keen [193] acknowledges four causes of inertia relative to organizational information systems. He indicates that information is only a small component, human information processing is

experiential and relies on simplification, organizational change is incremental and evolutionary with large changes being avoided, and that data is a political resource affecting particular groups as well as an intellectual commodity. Each of these suggests the importance of a knowledge of the way in which information is processed by organizations.

Of particular interest among studies concerning information processing in organizations are the works of Baron [28], Ebert and Mitchell [89], Fick and Sprague [110], Gerwin and Tuggle [129], Howell and Fleishman [165], Huber [169]-[171], Keen [193], Libby and Lewis [215], Lucas [225]-[228], O'Reilly [268], Shumway *et al.* [329], Simon [342], Starbuck and Nystrom [357], Taggart and Tharp [367], Tushman and Nadler [379], Tuggle and Gerwin [380], Wright [406], and Zedeck [409].

The purpose of systems for planning and decision support is to provide timely, relevant, and accurate information to system users such as to enhance human judgment, and decisionmaking efficiency and effectiveness concerning resource allocations that affect issues under consideration. To enhance efficiency and effectiveness available resources must be allocated and coordinated in space, across a hierarchy of decisionmakers; and in time, as new information arrives and the environmental situation extant changes. Associated information acquisition, analysis, and evaluation and interpretation must, as a consequence, often be distributed both in space and in time. This must be accomplished selectively in space and time since different decisionmakers have different information needs. In addition, it will be physically impossible and often behaviorally undesirable to supply all relevant information to each decisionmaker in the hope that it will be effectively cognized and utilized. Further, differences in education, motivation, experiences with the environmental situation extant, and stress will influence cognitive information processing style. Consequently a central task in the design of effective information systems is that of selection and choice of appropriate information system architecture to enhance selective information processing in order to provide each user of the system with the most appropriate information at the most appropriate time. Thus questions of information selection, information aggregation in space and in time, and the contingency task structure become of major importance.

It is desirable that an appropriately designed system, and the associated process, be capable of the following.

1) Assisting in the evaluation of alternative plans and courses of action that involve formal operational thought processes.
2) Assisting in the transfer of formal operational situations to concrete operational situations.
3) Assisting in evaluation of alternative plans and courses of action that involve concrete operational thought processes.
4) Assisting in the avoidance of information processing biases and poor judgmental heuristics.
5) Assisting in the proper aggregation of information cues from multiple distributed sources.
6) Assisting in the use of a variety of judgmental heuristics appropriate for given operational environments as natural extensions of a decisionmaker's normal cognitive style.
7) Assisting, to the extent possible, in the determination of whether a formal or concrete style of cognition is most appropriate in a given situation.
8) Assisting decisionmakers who need to use formal operational thought and those whose expertise allows appropriate and effective use of concrete operational thought to function together in a symbiotic and mutually supportive way.

Clearly there is a space-time and an organizational dependence associated with these desired capabilities. Among the many concerns that dictate needs and requirements for automated support systems is the fact that decisionmakers must typically make more judgments and associated decisions in a given period of time than they can comfortably make. This creates a stressful situation which can lead, as has been noted, to the use of poor information processing and judgmental heuristics, especially since judgments and decisions are typically based on forecasts of the future and, therefore, inherently involve uncertainty.

There are formidable needs and issues to be resolved that are associated with the design of information processing and judgment aiding support systems. These relate to questions concerning appropriate functions for the decisionmaker and staff to perform. They concern the type of information which should be available and how this information should be acquired, analyzed, stored, aggregated, and presented such that it can be used most effectively in a variety of potential operational environments. They concern design of information systems with strong space-time-environmental dependencies. They concern design of information systems that can effectively "train" people to adapt and use appropriate concrete operational heuristics in those environments in which inexperience dictates initial use of formal operational thought. They concern design and use of information systems that support environmentally experienced decisionmakers in the use of a variety of effective concrete operational heuristics. And because of their use by multiple decisionmakers, these tasks must be accomplished in a parallel architectural fashion.

Huber [170]-[171] and Tushman and Nadler [379] have developed a number of propositions, based on their own research and upon the research of others, reflecting various aspects of information processing in organizations. There are a number of fundamental propositions developed by Tushman and Nadler which relate to the development of a model of an organization as an information processing system. These fundamental propositions include [379] the following.

- FP1: Tasks of organizations and their subunits vary in uncertainty and risk variables.
- FP2: As uncertainty and risk increase so also does the need for information and increased information processing capability.
- FP3: Capacities and capabilities in information processing will vary as a function of organization structure.
- FP4: Organizational effectiveness increases as the match between information processing requirements and information processing capacity increases.
- FP5: Effectiveness of organizational units will depend upon their ability to adapt their internal structures over time to meet the changed information processing requirements that will be associated with environmental changes.

In an effort to enhance efficiency, organizational information processing typically requires selective routing of messages and summarization of messages. Huber [171] identifies six variables associated with the routing of messages. Six propositions relative to message routing are identified and associated with these variables. Three propositions are associated with delay in messages, eight with organizational message modification, and four with message summarization. Table II presents an interpretation of the impacts of the variables associated with organizational information processing and the probabilities of routing, delay, modification, and summarization of messages. It is possible to infer a few impacts not discussed in this noteworthy work of Huber. Most of these simply relate to the observation that if something happens to decrease the probability of sending a message unmodified then the probability of the message being delayed and/or modified is increased.

Identification of other variables which influence information processing by organizations would represent a desirable activity. To determine how these information processing variables are influenced by the information processing biases of individuals discussed in Section III would seem especially desirable in terms of the likely usefulness of the results and the need for an expanded theory of group information processing biases. There

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

DESIGN OF INFORMATION SYSTEMS AND PROCESSES

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

TABLE II
CROSS IMPACT MATRIX BETWEEN VARIABLES AFFECTING ORGANIZATIONAL INFORMATION AND ASSOCIATED ACTIVITIES

	(a) PROBABILITY OF MESSAGE ROUTING TO UNDESIRED, UNSUMMARIZED POINT	(b) PROBABILITY OF MESSAGE DELAY	(c) PROBABILITY OF EXTENT OF MESSAGE MODIFICATION	(d) PROBABILITY OF EXTENT OF MESSAGE SUMMARIZATION
1. INCREASES IN ECONOMIC AND OTHER COSTS IN A TRANSMISSION SENDING	−	⊖	−	−
2. INCREASES IN WORKLOAD OF SENDING UNIT	−	+	+	−
3. PERCEIVED RELEVANCE OF MESSAGE TO SENDING UNIT	+	⊖	⊖	+
4. DECREASES IN PERCEIVED GOAL ATTAINMENT, STATUS OR POWER OF THE SENDING UNIT RESULTING FROM ROUTING	−	⊕	⊕	
5. INCREASES IN PERCEIVED GOAL ATTAINMENT, STATUS OR POWER RESULTING FROM MODIFICATION	⊖		+	
6. PERCEIVED GOAL ATTAINMENT, STATUS, OR POWER OF THE SENDING UNIT	+	⊖	⊖	
7. FREQUENCY OF PAST COMMUNICATION OF SIMILAR MESSAGES	+	⊖	⊖	
8. PERCEIVED TIMELINESS OF MESSAGE FOR THE RECEIVING UNIT		−		
9. NUMBER OF ACTIVE COMMUNICATION LINKS IN THE CHAIN BETWEEN RECEIVER AND SENDER	⊖	+	+	+
10. DECREASE IN STATUS OF THE RECEIVER PERCEIVED BY THE SENDER TO RESULT FROM MODIFICATION	⊖		+	
11. AMOUNT OF DISCRETION ALLOWED ALTERING OR CHOOSING THE MESSAGE FORMAT	⊖		+	
12. INCREASED INDIFFERENCE BETWEEN ACTUAL MESSAGE CONTENT AND TRANSMITTER'S DESIRED CONTENT	⊖	⊕	+	
13. INCREASED IN PERCEIVED AMBIGUITY OF DATA ON WHICH MESSAGE IS BASED	⊖	⊕	+	
14. INCREASES IN SAVINGS DUE TO SUMMARIZATION				+

\+ = ENHANCING IMPACT SEEN BY [17]
− = INHIBITING IMPACT SEEN BY [17]
⊕ = INFERRED ENHANCING IMPACT
⊖ = INFERRED INHIBITING IMPACT

appears to have been only limited results obtained in the area of cognitive information processing biases and use of inferior heuristics on the part of groups. Thus many of the areas discussed in Sections III and IV could be extended to groups

Especially noteworthy concerning results that have been obtained in this area are the groupthink studies of Janis and Mann [177]. Groupthink is a collective pattern of defensive avoidance, a concurrence-seeking tendency of highly cohesive groups. When groupthink occurs, people develop rationalizations to support selectively perceived illusions or wishful thinking about issues at
nd and typically participate collectively in development of a
ensive avoidance pattern. In groupthink a group collectively falls victim to one or more of the cognitive biases described in Section III.

Among the conditions which lead to groupthink are high cohesiveness, insulation, lack of use of systemic procedures for

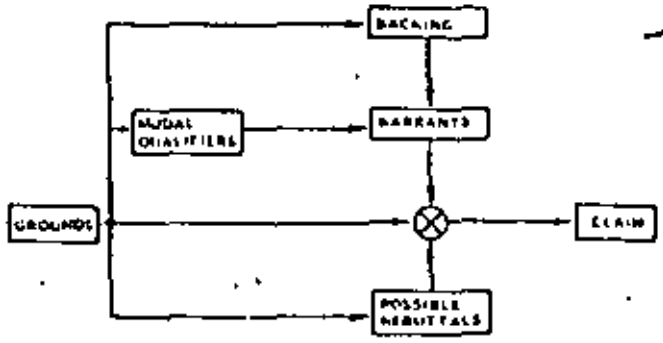


Fig. 11. A possible structure for information processing based upon the six elements of logical reasoning.

search and appraisal, highly directive leadership, and a contingency task situation which leads to high stress. Among the symptoms of groupthink cited by Janis and Mann [177] are an illusion of invulnerability, collective rationalization, belief in inherent group morality, excessive pressure against dissenting views, self-censorship, illusions of unanimity, and members who shield the group from disconfirming information. They cite a large number of case studies involving groupthink; cases where incrementalism and bureaucratic politics were the dominant decisionmaking framework typically adopted. Nine prescriptions are offered to avoid groupthink.

1) The group leadership should be noncommitted to particular alternative courses of action.
2) The group leader should encourage critical evaluation.
3) "Devil's advocates" should be included in the group.
4) Subgroups should be formed, allowed to function independently, and then meet with other subgroups to express generated ideas and resolve differences.
5) A variety of alternative scenarios of potential opponents intentions should be developed.
6) Second-opinion meetings should be held to allow full expression of doubts and rethinking of the issue.
7) Experts with opposite viewpoints to the majority view should be encouraged to present challenging views.
8) A small "policy" subgroup should always discuss subgroup deliberation with the larger group to attempt to obtain disconfirming feedback.
9) Independent policy planning and evaluation groups should be formed.

The suggestions offered in Section III to avoid cognitive bias and to ameliorate the effects of those that do occur appear capable of application to groups as well as to individuals. Explicit study of group and organizational bias that would compliment and extend existing studies [16], [19], [129], [155], [173], [257], [276]–[279], [280], [388], [390], [391], [394] of group and organizational decisionmaking should yield results that are valuable for the design of planning and decision support systems.

A major difficulty in cognitive information processing seems to be failure to identify and use an appropriate structure that allows appropriate weighting of observed data. Investigation of the effects of various structured information processing/decision aiding protocols upon the acquisition, analysis, and interpretation of information and its integration with judgment and decisionmaking activities would appear to be a contemporary need in information system design. There are six elements found in explicit argument [376]

1) *claims* or hypotheses,
2) *grounds* or foundations to support the claims,
3) *warrants* or justification for the grounds or foundations,
4) *backing* or the general body of information that is presupposed by the warrant,

TABLE III

INTERPRETATION OF THE JANIS AND MANN [177] COPING PATTERNS

Janis and Mann Coping Pattern	Likely information-processing characteristics	Level of interest and importance attached to information & decision task	Stress level	Information evaluation style	Potential coping errors	Characteristics of possible errors	Characteristics of proper selection of coping pattern
Unconflicting adherence or unconflicting change	Indifference to information acquisition and analysis	Low	Low, calm consumer	Concrete operational	Lack of interest in information processing has caused a generally unstructured and unfamiliar issue to be displayed as a familiar well structured, or an inconsequential, one.	An unconcerned tranquil journey to disaster	efficient and effective use of past experience to select an appropriate decision
Defensive avoidance	May vary from no difference to information, to highly selective processing of information to avoiding disconfirming information, thereby encouraging "wishful thinking".	Low to medium	Highly variable from low to high	Generally flawed and concrete operational, inattention, delay, and refusal to evaluate and act.	Decisions are postponed, procrastinated, or avoided by shifting responsibility to others and ignoring essentially all information.	No decision is, in reality, a decision. One of the possible consequences of no decision will occur.	This is never a proper coping pattern; only "going back" will result in a good outcome.
					A form of wishful thinking is used to to evolve a decision. Highly selective perception is used to bolster the decision.	Often value liabilities and inconsistencies will result in accepting an hypothesis that the decisionmaker believes is deserving to be true as a self fulfilling prophecy.	
Hypervigilance	Very undiscriminate and erratic search	Very high	Very high	Flawed and former operational	Failure in acquisition and analysis of information resulting from information overload due to panic.	Highly variable from snap judgments and acceptance of overly simplified and flawed decision rules to freezing such that valuable time, in which a good decision could be made, is lost.	This is never a proper coping pattern; only "going back" will result in a good outcome.
Vigilance	Deemphasized discriminating information acquisition and analysis. Typically this involves: a) thorough landscape canvas of potential alternatives b) identification of all relevant objectives c) thorough impact analysis d) intensive search for new information e) unbiased cognizing of disconfirming as well as confirming information f) sensitivity analysis to parameter changes g) detailed implementation provisions and contingency planning	High	Moderate	Formal operational	The issue is incorrectly structured or not within the experiential background of the decisionmaker.	Wasted time and possibly an inferior decision to one that could have been obtained from effort of an analogous reasoning or other concrete operational thought.	Deemphasized discriminating search and deliberation involving an unstructured selection to select an appropriate decision.

5) *modal qualifiers* or circumstances, ... [illegible] ...ncies, or restrictions which will have to exist in ord... the warrant truly supports the grounds,
6) *possible rebuttals* or circumstances, ... [illegible] ...ncies, or restrictions which, if they exist, will refute o... [illegible] ...nish the force of the warrant.

A simplified block diagram of the interaction among these elements is given in Fig. 11. The information processing "structure," consisting in part of the decisionmakers' view of possible and probable action courses and the "decision situation model," is specified by elements 3–6. Element 2, the "grounds," comprises the situational data pertaining to the operational conditions extant. The claim, element 1, is the empirical statement which is supported by other elements in this information processing structure. Toulmin shows through examples that the six elements for logical argument and reasoning can be used as a model for rational reasoning in a number of areas including law, science, the arts, management, and ethics. This structured information processing model is also sufficiently general to accommodate analytical hierarchical inference [165], [306]. Thus it may well provide a structured framework for information processing that can accommodate a variety of information processing styles and approaches ranging from the purely qualitative and affective, to reasoning by analogy which may be a blend of qualitative and quantitative, to quantitatively based filtering and detection algorithms. This could provide enhanced understanding of the implications of chance, cause, and reason [55] in human information processing.

Use of a structured information processing format may reduce the tendency for message distortion due to the exacerbating variables presented in Table III, perhaps to a considerable extent. Mitroff and Mason [255] have presented some suggestions concerning use of structured logical reasoning to cope with ill-structured policy problems and the often occurring divergence between opposing formulations and perceptions of large-scale issues.

Information summarization is needed in information systems for a variety of reasons. Procedures to condense and organize information into a form that can be managed and used in an efficient manner are, therefore, important. The structured information processing model suggested here may well provide organizational support for message aggregation and integration that will accommodate and encourage effective information summarization. We postulate that this framework may accommodate both receptive and preceptive styles of processing and summarization of information, that it will also accommodate nonnumerical and numerical information; and thus hopefully enable rapid conversion from one to the other as needed or desired for different situations.

In this section we have examined a number of frameworks for decisionmaking. Our particular interest is in the description of these frameworks in a way compatible with and supportive of the effective design of systems and processes to aid groups in planning and decisionmaking. We describe a number of "rational" ways in which groups make decisions and pay particular attention to information processing needs in group decisionmaking. Use of a structured protocol for information processing in systems for planning and decision support is suggested as a generic suggestion of potential ways to detect and correct possible cognitive biases that affect many judgment tasks.

Acknowledgment

Helpful comments were provided by several people including Dr. Chelsea C. White, III, Adolfo Lagomasino, Elbert White, and Ranju Rao.

References

[1] G. Ahoovi and N. Howard, "A Boolean approach to ... program planning," *Management Sci.*, vol. 26, no. 7, pp. 719–735, July 1980.
[2] L. Adelman, T. R. Stewart, and K. R. Hammond, "A case history of the application of social judgment theory to policy formulation," *Policy Sciences*, vol. 6, pp. 137–159, 1975.
[3] M. Allais and O. Hagen, Eds., *Expected Utility Hypotheses and the Allais Paradox*. Boston, MA: D. Reidel Publishing Co., 1979.
[4] G. T. Allison, *Essence of Decision*. Boston, MA: Little, Brown, 1971.
[5] S. L. Alter, *Decision Support Systems: Current Practice and Continuing Challenge*. Reading, MA: Addison-Wesley, 1980.
[6] M. E. Ames, *Outcome Uncertain*. Washington, D.C.: Communications, 1978.
[7] J. R. Anderson, *Cognitive Psychology and Its Implications*. San Francisco, CA: W. H. Freeman and Co., 1980.
[8] N. H. Anderson and G. R. Alexander, "Choice Test of the Averaging Hypothesis for Information Integration," *Cognitive Psychol.*, vol. 2, pp. 313–324, 1971.
[9] N. H. Anderson, "Cognitive algebra: Integration theory applied to social attribution," *Exp. Social Psychol.*, Academic, vol. 7, pp. 1–101, 1974.
[10] N. H. Anderson and J. Shanteau, "Weak inference with linear models," *Psychol. Bull.*, vol. 84, no. 6, pp. 1155–1170, 1977.
[11] N. H. Anderson, "Progress in cognitive algebra," in *Cognitive Theories in Social Psychology*, L. Berkowitz, Ed. New York: Academic, 1978.
[12] N. H. Anderson, "Algebraic rules in psychological measurement," *Amer. Scientist*, vol. 67, no. 5, pp. 555–563, Sept./Oct. 1979.
[13] H. I. Ansoff, "The state of practice in planning systems," *Sloan Management Rev.*, vol. [illegible], pp. 1–24, Winter 1977.
[14] J. S. Armstrong, *Long-Range Forecasting: From Crystal Ball to Computer*. New York: [illegible]ley, 1978.
[15] _____, "Forecasting with econometric methods: Folklore versus fact," *J. Bus.*, vol. 51, pp. 549–564, 1978.
[16] _____, "The seersucker theory: The value of experts in forecasting," *Tech. Rev.*, pp. 19–[illegible], June/July 1980.
[17] K. J. Arrow, *Social Choice and Individual Values*, 2nd ed. New Haven, CT: Yale Univ., 1963.
[18] W. Ascher, *Forecasting: An Appraisal for Policymakers and Planners*. Baltimore, MD: Johns Hopkins Univ., 1978.
[19] M. Bacharach, "Group decisions in the face of differences of opinion," *Management Sci.*, vol. 22, no. 2, pp. 182–191, Oct. 1979.
[20] R. F. Baker, R. M. Michelsen, and F. S. Preston, *Public Policy Development-Linking the Technical and Political Processes*. New York: Wiley, 1975.
[21] W. M. Balke, K. R. Hammond, and G. D. Meyer, "An alternative approach to labor-management negotiations," *Admin. Sci. Quart.*, vol. 18, pp. 311–327, 1973.
[22] D. P. Ballou and J. S. Pipkin, "Competitive strategies: A conjoint choice model," *Omega*, vol. 8, no. 1, pp. 53–62, 1980.
[23] R. L. Bankers and S. C. Gupta, "A process for hierarchical decision making with multiple objectives," *Omega*, vol. 8, no. 2, pp. 137–149, 1980.
[24] M. L. Barnett and E. J. Lusk, "Cognitive and personality factors for the design of management information systems," *Management Sci.*, vol. 23, no. 8, pp. 820–829, 1977.
[25] M. Bar-Hillel, "The base rate fallacy in probability judgments," *Acta Psychol.*, vol. 44, pp. 211–233, 1980.
[26] D. P. Baron, "Investment policy, optimality and the mean-variance model," *J. Finance*, vol. 34, no. 1, pp. 207–232, Mar. 1979.
[27] F. H. Barron, "Behavioral decision theory: A topical bibliography for management scientists," *Interfaces*, vol. 5, no. 1, pp. 56–62, Nov. 1974.
[28] _____, "An information processing methodology for [illegible]ing into decision processes," in *The Role of Effectiveness of Theories of Decision in Practice*, D. J. White and O. C. Bowen, Eds. New York: Crane, Russak and Co., Inc., 1975, pp. 195–206.
[29] F. H. Barron and B. B. Person, "Assessment of multiplicative utility functions via holistic judgments," *Organ. Behav. Human Perform.*, vol. 24, pp. 147–166, 1979.
[30] F. H. Barron and R. John, "Reference effects: A cheap suit's clothing," *Organ. Behav. and Human Perform.*, vol. 25, pp. [illegible]74, 1980.

[31] B. H. Beach, "Expert judgment about uncertainty: Bayesian decision making in realistic settings," *Organ. Behav. and Hum. Performance*, vol. 14, pp. 10–59, 1975.

[32] L. R. Beach and T. R. Mitchell, "A contingency model for the selection of decision strategies," *Academy of Management Rev.*, vol. 3, pp. 439–448, July 1978.

[33] D. E. Bell and H. Raiffa, "Marginal value and intrinsic risk aversion," June 1979 and "Decision regret: A component of risk aversion," Harvard Business School manuscripts, Harvard Univ., Cambridge, MA, June 1980.

[34] I. Benbasat and R. G. Schroeder, "An experimental investigation of some MIS design variables," *Management Inform. Syst. Quar.*, vol. 1, pp. 37–50, Mar. 1977.

[35] I. Benbasat and R. N. Taylor, "The impact of cognitive styles on information system design," *Management Inform. Syst. Quar.*, vol. 2, pp. 43–54, June 1978.

[36] J. L. Bennett, Ed., *Building Decision Support Systems*. Reading, MA: Addison-Wesley, 1981.

[37] J. R. Bettman, *An information processing theory of consumer choice*. Reading, MA: Addison-Wesley, 1979.

[38] T. R. Blakeslee, *The Right Brain*. New York: Anchor, 1980.

[39] R. H. Bonczek, C. W. Holsapple and A. B. Whinston, "Computer-based support of organizational decision making," *Decision Sci.*, vol. 10, pp. 268–291, 1979.

[40] _____, "The evolving roles of models in decision support systems," *Decision Sci.*, vol. 11, no. 2, pp. 337–356, Apr. 1980.

[41] _____, *Foundations of Decision Support Systems*. New York: Academic, 1981.

[42] E. Borgida and R. E. Nisbett, "Differential impact of abstract versus concrete information of decisions," *J. of Applied Social Psych.*, vol. 7, no. 3, pp. 258–271, 1977.

[43] C. J. Brainerd, *Piaget's Theory of Intelligence*. Englewood Cliffs, NJ: Prentice-Hall, 1978.

[44] D. Braybrooke and C. E. Lindblom, *A Strategy of Decision-Policy Evaluation as a Social Process*. New York: Free Press, 1970.

[45] B. Brehmer, "Social Judgment Theory and the Analysis of Interpersonal Conflict," *Psychol. Bull.*, vol. 83, pp. 985–1003, 1976.

[46] _____, "Response consistency in probabilistic inference tasks," *Organ. Behav. Hum. Perform.*, vol. 22, pp. 103–115, 1978.

[47] _____, "In one word: Not from experience," *Acta. Psychol.*, vol. 45, pp. 223–241, 1980.

[48] H. S. Bronkers, Ed., *Decision Making: Creativity Judgment and Systems*. Columbus, OH: Ohio State Univ., 1972.

[49] D. W. Broadbent, *Decision and Stress*. London: Academic, 1971.

[50] H. I. Brown, "On being rational," *J. Philos. Quar.*, vol. 15, pp. 41–48, 1978.

[51] R. V. Brown, A. S. Kahr, and C. Peterson, *Decision Analysis for the Manager*. New York: Holt, Rinehart, and Winston, 1974.

[52] W. A. Buehring, W. K. Fish, and R. L. Keeney, "Examining energy/environment policy using decision analysis," *Energy Syst. and Policy*, vol. 2, no. 3, pp. 341–367, 1979.

[53] D. W. Bunn, "Policy analytic implications for a theory of prediction and decision," *Policy Sci.*, vol. 8, pp. 125–134, 1977.

[54] _____, "Screening methods in policy analysis," *Socio-Econ. Planning Sci.*, vol. 12, pp. 329–331, 1978.

[55] A. W. Burks, *Chance, Cause, Reason*. Chicago, IL: Univ. of Chicago, 1977.

[56] J. S. Carroll, "Analyzing decision behavior: The magicians' audience" in T. S. Wallsten, Ed., *Cognitive Processes in Choice and Decision Behavior*. Hillsdale, NJ: Erlbaum Ass., 1980, pp. 69–76.

[57] N. L. Chervany and G. W. Dickson, "On the validity of the analytic-heuristic instrument utilized in the Minnesota experiments—A reply," *Management Sci.*, vol. 24, pp. 1091–1092, 1978.

[58] R. W. Chesha and J. L. New, "Information support for decision-maker learning in a competitive environment: An experimental study," *Decision Sci.*, vol. 11, no. 4, pp. 603–615, Oct. 1980.

[59] J. Christensen-Szalanski, "Problem solving strategies: A selection mechanism, some implications, and some data," *Organ. Behav. Hum. Perform.*, vol. 23, pp. 307–323, Oct. 1978.

[60] _____, "A further examination of the selection of problem-solving strategies: The effects of deadlines and analytic aptitudes," *Organ. Behav. Hum. Perform.*, vol. 25, pp. 107–122, 1980.

[61] L. J. Cohen, *The Probable and the Provable*. Clarendon, England: Oxford Univ., 1979.

[62] _____, "On the psychology of prediction: Whose is the fallacy," *Cognition*, vol. 7, no. 4, pp. 385–407, 1979.

[63] M. D. Cohen, J. G. March, and J. P. Olsen, "A garbage can model of organizational choice," *Admin. Sci. Quar.*, vol. 17, no. 1, pp. 1–25, 1972.

[64] R. L. Cook, and T. R. Stewart, "A comparison of seven methods for obtaining subjective descriptions of judgmental policy," *Organ. Behav. Hum. Perform.*, vol. 13, pp. 31–45, 1975.

[65] C. H. Coombs and D. G. Pruitt, "Components of risk in decision making: Probability and variance preferences," *J. Experimental Psychol.*, vol. 60, no. 5, pp. 265–277, Dec. 1960.

[66] C. H. Coombs and G. S. Avrunin, "Single-peaked functions and the theory of preference," *Psychol. Rev.*, vol. 84, pp. 216–230, 1977.

[67] F. I. M. Craik, "Human memory," *Ann. Rev. Psychol.*, vol. 30, pp. 63–102, 1979.

[68] R. M. Cyert and J. G. March, *A Behavioral Theory of the Firm*. Englewood Cliffs, NJ: Prentice-Hall, 1963.

[69] J. H. Davis, "Group decision and social interaction: A theory of social decision schemes," *Psychol. Rev.*, vol. 80, no. 2, pp. 97–125, Mar. 1973.

[70] R. M. Dawes, "The mind, the model, and the task," in *Cognitive Theory*, vol. 1, F. Restle *et al.*, Eds. Hillsdale, NJ: Lawrence Erlbaum Ass., 1975, pp. 119–130.

[71] _____, "The robust beauty of improper linear models in decision making," *Amer. Psychol.*, vol. 34, no. 7, pp. 571–582, July 1979.

[72] R. H. Day, Ed., *Adaptive Economics*. New York: Academic, 1975.

[73] H. D. Delaney and T. S. Wallsten, "Probabilistic information processing: Effects of a biased payoff matrix on choices and bids," *Organ. Behav. Hum. Perform.*, vol. 20, no. 2, pp. 203–237, Dec. 1977.

[74] M. DeWaele, "Managerial style and the design of decision aids," *Omega*, vol. 6, no. 1, pp. 5–13, 1978.

[75] A. DeWispelare and A. P. Sage, "On combined multiple objective optimization theory and multiple attribute utility theory for evaluation and choicemaking," *Large Scale Systems*, vol. 2, no. 1, pp. 1–19, 1981.

[76] G. W. Dickson, J. A. Senn, and J. L. Chervany, "Research in management information systems: The Minnesota experiments," *Management Sci.*, vol. 23, pp. 913–923, 1977.

[77] P. Diesing, *Reason in Society*. Urbana, IL: Univ. Illinois, 1962.

[78] R. H. Doktor and W. F. Hamilton, "Cognitive styles and the acceptance of management science recommendations," *Management Sci.*, vol. 19, no. 9, pp. 884–894, 1973.

[79] R. Doktor, "Problem solving styles of executives and management scientists," *TIMS Studies in the Management Sciences*, vol. 8, pp. 123–134, 1978.

[80] A. W. Drake, R. L. Keeney and P. M. Morse, *Analysis of Public Systems*. Cambridge, MA: Massachusetts Institute Tech., 1972.

[81] H. L. Dreyfus, *What Computers Can't Do: The Limits of Artificial Intelligence*. New York: Harper and Row, 1979.

[82] S. E. Dreyfus and H. L. Dreyfus, "A five stage model of the mental activities involved in directed skill acquisition," Univ. Calif. at Berkeley, Rep. ORC 80-2, Feb. 1980.

[83] M. J. Driver and T. J. Mock, "Human information processing, decision theory style, and accounting information systems," *Accounting Rev.*, vol. 50, pp. 490–508, 1975.

[84] Y. Dror, *Public Policymaking Reexamined*. San Francisco, CA: Chandler, 1978.

[85] _____, *Design for Policy Sciences*. New York: Elsevier, 1971.

[86] N. M. Duffy and M. G. Assad, *Information Management*. Capetown, South Africa: Oxford Univ., 1980.

[87] M. D. Dunnette, ed., *Handbook of Industrial and Organizational Psychology*. Chicago, IL: Rand-McNally, 1976.

[88] B. K. Dutta and W. R. King, "A competitive scenario modeling system," *Management Sci.*, vol. 26, no. 3, pp. 261–273, Mar. 1980.

[89] R. J. Ebert and T. R. Mitchell, *Organizational Decision Processes*. New York: Crane Russak and Co., 1975.

[90] H. Einhorn, "Use of nonlinear, noncompensatory models as a function of task and the amount of information," *Organ. Behav. Hum. Perform.*, vol. 5, pp. 1–27, 1971.

[91] H. J. Einhorn and R. M. Hogarth, "Unit weighting schemes for decision making," *Organ. Behav. Hum. Perform.*, vol. 13, pp. 171–192, 1975.

[92] _____, and E. Klempner, "Quality of group judgment," *Psychol. Bull.*, vol. 84, no. 1, pp. 158–172, 1977.

[93] H. J. Einhorn and W. McCoach, "A simple multi attribute utility procedure for evaluation," *Behav. Sci.*, vol. 22, pp. 270–282, 1977.

[94] H. J. Einhorn and R. M. Hogarth, "Confidence in judgment

Persistence of the illusion of validity," *Psychol. Rev.*, vol. 85, no. 5, pp. 395–416, 1978.
[95] H. J. Einhorn, D. N. Kleinmuntz, and B. Kleinmuntz, "Linear regression and process tracing models of judgment," *Psychol. Rev.*, vol. 86, no. 5, pp. 465–485, 1979.
[96] H. J. Einhorn, "Learning from experience and suboptimal rules in decision making," in T. S. Wallsten, Ed., *Cognitive Processes in Choice and Decision Behavior*. Hillsdale, NJ: Lawrence Erlbaum Ass., 1980, pp. 1–20.
[97] ———, "Overconfidence in judgment" in *Fallible Judgment in Behavioral Research*, R. A. Shweder, Ed. San Francisco, CA: Jossey Bass, 1980, pp. 1–16.
[98] H. J. Einhorn and R. M. Hogarth, "Behavioral decision theory: Processes of judgment and choice," *Annu. Rev. Psychol.*, vol. 32, pp. 53–88, 1981.
[99] K. A. Ericsson and H. A. Simon, "Verbal reports as data," *Psychol. Rev.*, vol. 87, no. 3, pp. 215–251, May 1980.
[100] W. K. Estes, *Handbook of Learning and Cognitive Processes*, vols. 1–6. Hillsdale, NJ: Lawrence Erlbaum Ass., 1975–1979.
[101] ———, "The cognitive side of probability learning," *Psychol. Rev.*, vol. 83, pp. 37–64, 1976.
[102] ———, "Is human memory obsolete? *Amer. Sci.*, vol. 68, pp. 62–69, 1980.
[103] A. Etzioni, "Mixed scanning, a 'third' approach to decision making," *Public Admin. Rev.*, vol. 27, pp. 385–392, Dec. 1967.
[104] ———, *The Active Society*. New York: Free Press, 1968.
[105] H. Eulau, "Problematics of decisional models in political contexts," *Amer. Behav. Scientist*, vol. 20, no. 1, pp. 127–144, Sept–Oct. 1976.
[106] P. H. Farquhar, "Advances in multiattribute utility theory," *Theory and Decision*, vol. 12, pp. 381–394, 1980.
[107] N. Feather, Ed., *Expectancy, Incentive, and Action*. Hillsdale, NJ: Lawrence Erlbaum Ass., 1981.
[108] P. A. Federico, K. E. Brun, and D. B. McCalla, *Management Information Systems and Organizational Behavior*. New York: Praeger, 1980.
[109] M. P. Feldman and A. Broadhurst, Eds., *Theoretical and Experimental Bases of Behavior Therapies*. London: Wiley, 1976.
[110] G. Fick, and R. H. Sprague, Jr., Eds., *Decision Support Systems: Issues and Challenges*. New York: Pergamon, 1980.
[111] G. Fischer, "Utility models for multiple objective decisions: Do they accurately represent human preferences," *Decision Sci.*, vol. 10, pp. 451–479, 1979.
[112] B. Fischhoff and R. Blyth, "I knew it would happen: Remembered probabilities of once future things," *Organ. Behav. Hum. Perfor.*, vol. 13, pp. 1–16, 1975.
[113] B. Fischhoff, "Hindsight and foresight: The effect of outcome knowledge on judgment under uncertainty," *J. of Exp. Psyc: Hum. Percep. and Perfor.*, vol. 1, no. 3, pp. 288–299, 1975.
[114] B. Fischhoff, P. Slovic, and S. Lichtenstein, "Knowing with certainty: The appropriateness of extreme confidence," *J. of Exp. Psyc: Hum. Percep. and Perfor.*, vol. 3, pp. 552–564, 1977.
[115] B. Fischhoff, "Fault trees: Sensitivity of estimated failure probabilities to problem presentation," *J. of Exp. Psyc: Hum. Percep. and Perform.*, vol. 4, pp. 342–355, 1978.
[116] ———, "For those condemned to study the past: Reflections on historical judgment," *Fallible Judgment in Behavioral Research*, R. A. Shweder, Ed. San Francisco, CA: Jossey Bass, 1980, pp. 79–84.
[117] B. Fischhoff, P. Slovic, and S. Lichtenstein, "Knowing what you want: Measuring labile values," *Cognitive Processes in Choice and Decision Behavior*, T. S. Wallsten, Ed. Hillsdale, NJ: Lawrence Erlbaum Ass., 1980, pp. 117–142.
[118] B. Fischhoff, "No man is a discipline," *Cognition, Social Behavior and the Environment*, J. Harvey, Ed. Hillsdale, NJ: Lawrence Erlbaum Ass., 1981.
[119] B. Fischhoff, B. Goitein, and Z. Shapira, "The experienced utility of expected utility approaches," *Expectancy, Incentive, and Action*, N. Feather, Ed. Hillsdale, NJ: Lawrence Erlbaum, Ass., 1981.
[120] M. Fishbein and I. Ajzen, *Belief, Attitude, Intention and Behavior*. Reading, MA: Addison-Wesley, 1975.
[121] P. C. Fishburn, "Lexicographic orders, utilities and decision rules: A survey," *Management Sci.*, vol. 20, no. 11, pp. 1442–1471, July 1974.
[122] ———, "Paradoxes of voting," *The Amer. Polit. Sci. Rev.*, vol. 68, pp. 537–546, 1974.
[123] ———, "Mean-risk analysis with risk associated with below target return," *Amer. Econ. Rev.*, vol. 67, pp. 116–126, 1977.
[124] P. C. Fishburn and R. G. Vickson, "Theoretical foundations of stochastic dominance" in *Stochastic Dominance*, G. A. Whitmore and M. G. Findlay, Eds. Lexington, MA: Heath, 1978, pp. 39–113.
[125] P. C. Fishburn, "On Handa's 'New theory of cardinal utility' and the maximization of expected return," *J. Polit. Econ.*, vol. 86, no. 2, pp. 321–324, Apr. 1978.
[126] J. H. Flavell, *Cognitive Development*. Englewood Cliffs, NJ: Prentice-Hall, 1977.
[127] D. L. Ford, H. Moskowitz, and D. R. Wittink, "Econometric modeling of individual and social multi-attribute utility functions," *Multivariate Behav. Res.*, vol. 13, no. 1, pp. 77–97, Jan. 1978.
[128] J. Fox, "Making decisions under the influence of memory," *Psychol. Rev.*, vol. 87, no. 2, pp. 190–211, 1980.
[129] D. Gerwin and F. D. Tuggle, "Modeling organizational decisions using the human problem-solving paradigm," *Academy of Management Rev.*, vol. 3, no. 4, pp. 762–773, 1978.
[130] M. L. Gick and K. J. Holyoak, "Analogical Problem Solving," *Cognitive Psychology*, vol. 12, pp. 306–355, 1980.
[131] H. Ginsburg and S. Opper, *Piaget's Theory of Intellectual Development*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1979.
[132] R. W. Goldsmith, "Studies of a model for evaluating judicial evidence," *Acta. Psychol.*, vol. 45, pp. 213–221, 1980.
[133] G. A. Gorry and M. S. S. Morton, "A framework for management information systems," *Sloan Management Rev.*, vol. 13, no. 1, pp. 55–70, 1971.
[134] T. B. Greene, S. M. Lee, and W. B. Newsome, Eds., *The Decision Science Process*. New York, Petrocelli, 1978.
[135] C. D. Hah and Lindquist, "The 1952 steel seizure revisited: A systematic study in presidential decision-making," *Admin. Sci. Quar.*, vol. 20, pp. 567–605, Dec. 1975.
[136] Y. Haimes, Ed., *Risk Benefit Analysis in Water Resources Planning and Management*. New York: Plenum, 1981.
[137] D. Hamilton, Ed., *Cognitive Processes in Stereotyping and Intergroup Perception*. Hillsdale, NJ: Lawrence Erlbaum Ass., 1981.
[138] J. S. Hammond (III), "Do's and don'ts of computer models for planning," *Harvard Business Review*, Harvard Univ., vol. 52, no. 2, pp. 110–113, Mar.–Apr. 1974.
[139] J. S. Hammond, "The roles of the manager and management scientist in successful implementation," *Sloan Management Rev.*, vol. 20, pp. 1–24, Winter 1979.
[140] K. R. Hammond, J. L. Mumpower, and T. H. Smith, "Linking environmental models with models of human judgment: A symmetrical decision aid," *IEEE Trans. Syst., Man., Cybern.*, vol. SMC-7, pp. 358–367, 1977.
[141] K. R. Hammond, Ed., *Judgment and Decision in Public Policy Formulation*. Boulder, CO: Westview, 1978.
[142] K. R. Hammond, G. H. McClelland, and J. Mumpower, *Human Judgment and Decision Making: Theories, Methods and Procedures*. New York: Hemisphere/Praeger, 1980.
[143] K. R. Hammond, "*The Integration of Research in Judgment and Decision Theory*," Univ. Colorado Institute of Behav. Sci., Boulder, Rep. CRJP 226, July 1980.
[144] J. Handa, "Risk, probabilities, and a new theory of cardinal utility," *J. of Polit. Econ.*, vol. 85, no. 1, pp. 97–122, Feb. 1977.
[145] J. C. Harsanyi, *Essays on Ethics, Social Behavior, and Scientific Explanation*. Boston, MA: D. Reidel, 1976.
[146] J. L. Harsanyi, *Rational Behavior and Bargaining Equilibrium in Games and Social Situations*. Cambridge, England: Cambridge Univer., 1977.
[147] J. C. Harsanyi, "Bayesian decision theory, rule utilitarianism, and Arrow's impossibility theorem," *Theory and Decision*, vol. 11, pp. 289–317, 1979.
[148] J. Haugeland, "The nature and plausibility of cognitivism," *Behav. Brain Sci.*, vol. 1, no. 2, pp. 215–260, Dec. 1978.
[149] J. R. Hauser, "Consumer preference axioms: Behavioral postulates for describing and predicting stochastic choice," *Management Sci.*, vol. 13, pp. 9404–9416, 1976.
[150] B. Hayes-Roth and F. Hayes-Roth, "A cognitive model of planning," *Cognitive Science*, vol. 3, pp. 275–310, 1979.
[151] J. C. Henderson and P. C. Nutt, "The influence of decision making style on decision making behavior," *Management Sci.*, vol. 26, no. 4, pp. 371–386, Apr. 1980.
[152] J. C. Hershey and P. J. Schoemaker, "Prospect theory's reflection hypothesis: A critical examination," *Organ. Behav. Hum. Perform.*, vol. 25, pp. 395–418, 1980.

[153] _____, "Risk taking and problem context in the domain of losses: An expected utility analysis," *J. of Risk and Insurance*, vol. 47, no. 1, pp. 111–132, 1980.

[154] P. J. Hoffman, T. C. Earle, and P. Slovic, "Multidimensional functional learning (MFL) and some new conceptions of feedback," *Organ. Behav., Hum. Perform.*, vol. 23, pp. 75–102, 1981.

[155] R. M. Hogarth, "A note on aggregating opinions," *Organ. Behav. Hum. Perform.*, vol. 21, pp. 121–129, 1978.

[156] _____, "Beyond discrete biases: Functional and dysfunctional aspects of judgmental heuristics," *Psychological Bulletin*, vol. 90, pp. 197–217, 1981.

[157] _____, "Judgment, drug monitoring and decision aids," *Monitoring for Drug Safety*. W. H. W. Inman, Ed. Lancaster, England: MTP, Ltd., pp. 439–475, 1980.

[158] R. M. Hogarth, C. Michaud, and J. L. Mery, "Decision behavior in urban development: A methodological approach and substantive considerations," *Acta. Psychologica*, vol. 45, pp. 95–117, 1980.

[159] R. M. Hogarth, *Judgment and Choice: The Psychology of Decision*. New York: Wiley, 1980.

[160] R. M. Hogarth and S. Makridakis, "The value of decision making in a complex environment: An experimental approach," *Management Sci.*, vol. 27, no. 1, pp. 93–107, Jan. 1981.

[161] _____, "Forecasting and planning: An evaluation," *Management Sci.*, vol. 27, no. 2, pp. 115–138, Feb. 1981.

[162] C. A. Holloway, *Decision making under uncertainty: Models and choice*. Englewood Cliffs, NJ: Prentice-Hall, 1979.

[163] C. A. Hooker, J. J. Leach, and E. F. McClennen, Eds., *Foundations and Applications of Decision Theory*, vols. I and II. Boston, MA: D. Reidel, 1978.

[164] R. A. Howard, J. E. Matheson, and R. E. Miller, *Readings in Decision Analysis*. Stanford Research Inst., Menlo Park, CA, 1976.

[165] S. C. Howell and E. A. Feldman, *Information Processing and Decision Making*. Hillsdale, NJ: Lawrence Erlbaum Ass., 1981.

[166] S. C. Howell and S. H. Lamont, "Uncertainty measurement: A cognitive taxonomy," *Organ. Behav. Hum. Perform.*, vol. 22, pp. 45–68, 1978.

[167] W. Howell, *Human Performance and Productivity*. Hillsdale, NJ: Lawrence Erlbaum Ass., 1981.

[168] O. Huber, "The influence of some task variables on cognitive operations in an information processing decision model," *Acta Psychol.*, vol. 45, no. 1–3, pp. 187–196, Aug. 1980.

[169] G. P. Huber, *Managerial Decision Making*. Glenview, IL: Scott, Foresman, 1980.

[170] _____, "Organizational information processing: Changes in the form and meaning of messages," Univ. Wisconsin, Working Paper 6-88-15, July 1980.

[171] _____, "Organizational information systems: Determinants of their performance and behavior," Univer. Wisconsin, Working Paper 4-81-7, Apr. 1981, *Management Science*, to be published.

[172] J. H. Huysman, "The effectiveness of the cognitive style construct in implementing operations research proposals," *Management Sci.*, vol. 17, pp. 92–104, 1970.

[173] A. Hylland and R. Zeckhauser, "The impossibility of Bayesian group decision making with separate aggregation of beliefs and values," *Econometrica*, vol. 47, no. 6, pp. 1321–1336, Nov. 1979.

[174] B. Ives, S. Hamilton and G. B. Davis, "A framework for research in computer based management information systems," *Management Sci.*, vol. 26, no. 9, pp. 910–934, Sept. 1980.

[175] A. G. Jago, "Configural cue utilization in implicit models of leader behavior," *Organ. Behav. Hum. Perform.*, vol. 22, no. 3, pp. 474–495, Dec. 1978.

[176] I. L. Janis, and L. Mann, "Coping with decisional conflict," *Amer. Sci.*, vol. 64, pp. 657–667, Nov.–Dec. 1976.

[177] _____, *Decision Making*. New York: Free Press, 1977.

[178] E. Jantsch, "*Technological Forecasting in Perspective*," Organ. for Econ. Cooperation. Dev., Paris, France, 1967, also *Perspectives on Planning*, Organ. for Econ. Cooperation. Dev., Paris, France, 1969.

[179] E. M. Johnson and G. P. Huber, "The technology of utility assessment," *IEEE Trans. Syst. Man. Cybern.*, vol. SMC-7, no. 5, pp. 311–325, May 1977.

[180] E. Johnson, *Deciding how I decide: The effect of making a decision*, Univ. Chicago, Chicago, IL, Center for Decision Res., Dec. 1979.

[181] H. Jungerman, "Speculations about decision-theoretic aids for personal decisions," *Acta Psychol.*, vol. 45, pp. 7–34, 1980.

[182] H. Jungerman and G. De Zeeuw, Eds., *Decision Making and Change in Human Affairs*. Boston, D. Reidel, 1977.

[183] D. Kahneman and A. Tversky, "On the psychology of prediction," *Psy. Rev.*, vol. 80, pp. 237–251, 1973.

[184] _____, "Prospect theory: An analysis of decisions under risk," *Econometrica*, vol. 47, no. 2, pp. 263–291, Mar. 1979.

[185] D. Kahneman, P. Slovic, and A. Tversky, Eds., *Judgment Under Uncertainty: Heuristics and Biases*. New York: Cambridge Univ., 1981.

[186] M. F. Kaplan and F. Schwartz, Eds., *Human Judgment and Decision Processes*. New York: Academic, 1975.

[187] _____, *Human Judgment and Decision Processes in Applied Settings*. New York, Academic, 1977.

[188] U. S. Karmarkar, "Subjectively weighted utility: A descriptive extension of the expected utility model," *Organ. Behav., Hum. Perform.*, vol. 21, no. 1, pp. 61–72, Feb. 1978.

[189] _____, "Subjectively weighted utility and the Allais paradox," *Organ. Behav. Hum. Perform.*, vol. 24, pp. 67–72, 1979.

[190] S. M. Kassin, "Consensus information, prediction and causal attribution: A review of the literature and issues," *J. Personality and Social Psychol.*, vol. 37, no. 11, pp. 1966–1981, 1979.

[191] D. L. Keefer and C. W. Kirkwood, "A multiobjective decision analysis: Budget planning for product engineering," *Oper. Res.*, vol. 29, no. 5, pp. 435–442, 1978.

[192] P. G. W. Keen and M. S. Morton, *Decision support systems: An organizational perspective*. Reading, MA: Addison-Wesley, 1978.

[193] P. G. W. Keen, "Information systems and organizational change," *Communications of the Assoc. for Computing Machinery*, vol. 24, no. 1, pp. 24–33, Jan. 1981.

[194] R. L. Keeney and C. W. Kirkwood, "Group decision making using cardinal social welfare functions," *Management Sci.*, vol. 22, no. 4, pp. 430–437, Dec. 1975.

[195] R. L. Keeney, "A group preference axiomatization with cardinal utility," *Management Sci.*, vol. 23, no. 2, pp. 140–145, Oct. 1976.

[196] R. L. Keeney and H. Raiffa, *Decisions with Multiple Objectives, Preferences and Value Tradeoffs*. New York: Wiley, 1976.

[197] R. L. Keeney, *Siting Energy Facilities*. New York: Academic, 1980.

[198] G. J. Kelleher, *The Challenge to Systems Analysis—Public Policy and Social Change*. New York, Wiley, 1970.

[199] H. H. Kelley and J. L. Michela, "Attribution theory and research," *Annu. Rev. Psychol.*, vol. 31, pp. 457–501, 1980.

[200] R. H. Kilmann, *Social Systems Design-Normative Theory and the MAPS Design Technology*. New York: North-Holland, 1977.

[201] C. W. Kirkwood, Ed., "Decision analysis special issue," *Oper. Res.*, vol. 28, no. 1, pp. 1–252, Jan./Feb. 1981.

[202] G. A. Klein and J. Weitzenfeld, "Improvement of skills for solving ill-defined problems," *Educ. Psychol.*, vol. 13, pp. 31–41, 1978.

[203] G. A. Klein, "Automated aids for the proficient decision maker," *Proc. 1980 IEEE Conf. Cybern., Soc.*, Boston, MA, pp. 301–304, Oct. 1980.

[204] D. N. Kleinmuntz and B. Kleinmuntz, "Decision strategies in simulated environments," *Behav. Sci.*, vol. 26, pp. 294–305, 1981.

[205] H. Kunreuther, "Limited knowledge and insurance protection," *Public Policy*, vol. 24, pp. 227–261, 1977.

[206] H. C. Kunreuther and P. Slovic, "Econ., Psychol., and Protective Behav," *Amer. Econ. Rev.*, vol. 68, no. 2, pp. 64–69, May 1978.

[207] H. C. Kunreuther and P. J. H. Schoemaker, "Decision analysis for complex systems: Integrating descriptive and prescriptive components," in *Knowledge: Creation, Diffusion, Utilization*, vol. 2, no. 3, pp. 389–412, Mar. 1981.

[208] F. Lad, "Embedding Bayes' theorem in general learning rules: Connections between idealized behavior and empirical research on learning," *British J., Math., Stat. Psychol.*, vol. 31, no. 2, pp. 113–125, Nov. 1978.

[209] E. J. Langer, "The illusion of control," *J. Personality Soc. Psychol.*, vol. 32, no. 2, pp. 311–328, 1975.

[210] E. J. Langer and J. Roth, "The effect of sequence of outcomes in a chance task on the illusion of control," *J. of Personality and Social Psychol.*, vol. 32, pp. 951–955, 1975.

[211] H. J. Leavitt, *Managerial Psychology*. Chicago, IL: Univ. Chicago, 1978.

[212] M. Leontiades and A. Tezel, "Planning perceptions and planning results," *Strategic Management J.*, vol. 1, pp. 65–75, 1980.

[213] R. Libby, "The use of simulated decision makers in information evaluation," *The Accounting Rev.*, vol. 50, pp. 475–489, July 1975.

[214] R. Libby and P. C. Fishburn, "Behavioral models of risk taking in business decisions: A survey and evaluation," *J. Accounting Res.*, vol. 15, no. 3, pp. 272–292, Autumn 1977.

[215] R. Libby and B. L. Lewis, "Human information processing research in accounting," *Accounting, Organizations and Society*, vol. 2, pp. 245–254, 1977.

[216] S. Lichtenstein and B. Fischhoff, "Do Those Who Know More Also Know More About How Much They Know?" *Organ. Behav. Hum. Perform.*, vol. 20, pp. 159–183, 1977.
[217] ———, "Training for Calibration," *Organ. Behav. Hum. Perform.*, vol. 25, pp. 149–171, 1980.
[218] C. E. Lindblom, "The science of 'muddling through'," *Public Admin. Rev.*, vol. 19, pp. 155–169, Spring 1959.
[219] ———, *The intelligence of democracy: Decision making through mutual adjustment*. New York: Free Press, 1965.
[220] ———, *Politics and Markets*. New York: Basic Books, 1977.
[221] ———, *The Policy Making Process*. Englewood Cliffs, NJ: Prentice-Hall, 1980.
[222] D. Lindley, *Making Decisions*. New York: Wiley, 1971.
[223] D. V. Lindley, A. Tversky, and R. Brown, "On the reconciliation of probability assessments," *J. Royal Statis. Soc., Series A*, vol. 142, part 2, pp. 146–180, 1979.
[224] J. D. C. Little, "Models and managers: The concept of a decision calculus," *Management Sci.*, vol. 16, no. 9, pp. B466–B485, Apr. 1970.
[225] H. C. Lucas, Jr., "A descriptive model of information systems in the context of organizations," *Data Base*, vol. 5, no. 2, pp. 27–36, 1973.
[226] ———, *Why information systems fail*. New York: Columbia Univ., 1975.
[227] ———, *Design, Implementation and Management of Information Systems*. New York: McGraw-Hill, 1976.
[228] H. C. Lucas, Jr., and N. R. Nielson, "The impact of the mode of information presentation on learning and performance," *Management Sci.*, vol. 26, no. 10, pp. 982–993, Oct. 1980.
[229] E. J. Lusk and M. Kersnick, "The effect of cognitive style and report format on task performance: The MIS design consequences," *Management Sci.*, vol. 25, no. 6, pp. 787–798, Aug. 1979.
[230] D. Lyon and P. Slovic, "Dominance of accuracy information and neglect of base rates in probability estimation," *Acta. Psychol.*, vol. 40, pp. 287–298, 1976.
[231] R. R. MacCrimmon and J. K. Siu, "Making trade-offs," *Decision Sci.*, vol. 5, pp. 680–705, 1974.
[232] K. R. MacCrimmon and R. N. Taylor, "Decision making and problem solving," *Handbook of Industrial and Organizational Psychology*, M. D. Dunnette, Ed. Chicago, IL: Rand McNally, ch. 32, pp. 1397–1453, 1976.
[233] A. J. MacKinnon and A. J. Wearing, "Complexity and decision making," *Behav. Sci.*, vol. 25, pp. 285–296, 1980.
[234] M. J. Mahoney and B. G. DeMonbreun, "Psychology of the scientist: An analysis of problem solving bias," *Cognitive Therapy and Res.*, vol. 1, pp. 229–238, 1977.
[235] S. Makridakis and S. C. Wheelwright, Eds., *Forecasting*. New York: North-Holland, 1979.
[236] J. G. March, "Bounded rationality, ambiguity, and the engineering of choice," *The Bell J. of Econ.*, vol. 10, pp. 587–608, 1978.
[237] J. G. March and J. P. Olsen, "Ambiguity and Choice in Organization," Universitetsfor., 1979.
[238] R. O. Mason and I. I. Mitroff, "A program for research on management information systems," *Management Sci.*, vol. 19, no. 5, pp. 475–485, 1973.
[239] R. O. Mason and E. B. Swanson, Eds., *Measurement for management decision*. Reading, MA: Addison-Wesley, 1981.
[240] A. M. McCosh and M. S. Morton, *Management Decision Support Syst.*, New York: Halsted, 1978.
[241] C. G. McGuire and R. Radner, *Decision and Organization*. Amsterdam: North-Holland, 1972.
[242] J. L. McKenney and P. G. W. Keen, "How managers' 'minds work'," *Harvard Bus. Rev.*, vol. 52, no. 3, pp. 79–90, May/June 1974.
[243] A. J. Meltsner, *Policy Analysts in the Bureaucracy*. Berkeley, Univer. Calif., 1976.
[244] J. A. Machea, EG. G. J. Elkman, and L. F. W. DeKlerk, *Handbook of Psychonomics*, vols. I and II. New York: North-Holland, 1979.
[245] C. Miller and A. P. Sage, "A methodology for the evaluation of research and development projects and associated resource allocation," *Comput., Elec. Eng.*, vol. 2, no. 2, pp. 123–152, June 1981.
[246] H. Mintzberg, "Planning on the left and managing on the right," *Harvard Bus. Rev.*, vol. 54, pp. 49–58, 1976.
[247] H. Mintzberg, D. Raisinghani, and A. Theoret, "The structure of unstructured decision processes," *Admin. Sci. Quar.*, vol. 21, pp. 246–275, June 1976.
[248] H. Mintzberg, "Structure in 5's: A synthesis of the research on organizational design," *Management Sci.*, vol. 26, no. 3, pp. 322–341, Mar. 1980.
[249] W. Mischel, "Toward a cognitive social reconceptualization of personality," *Psychol. Rev.*, vol. 80, pp. 250–283, 1973.
[250] ———, *Personality Assessment*. New York: Wiley, 1950.
[251] ———, "On the interface of cognition and personality: Beyond the person–situation debate," *Am. Psychol.*, vol. 34, pp. 740–754, 1979.
[252] I. Mitroff, J. Nelson, and R. O. Mason, "On management myth information systems," *Management Sci.*, vol. 21, no. 4, pp. 371–382, Dec. 1974.
[253] I. I. Mitroff and R. H. Kilmann, "Stories managers tell: A new tool for organizational problem solving," *Management Rev.*, vol. 64, no. 7, pp. 18–29, July 1975.
[254] I. I. Mitroff and R. H. Kilmann, *Methodological approaches to social science*. San Francisco, CA: Jossey Bass, vol. 34, no. 9, Sept. 1979.
[255] I. I. Mitroff and R. O. Mason, "Structuring ill-structured policy issues: Further explorations in a methodology for messy problems," *Strategic Management*, vol. 1, pp. 331–342, 1980.
[256] H. Montgomery and O. Svenson, "On decision rules and information processing strategies for choices among multiattribute alternatives," *Scandinavian J. Psychol.*, vol. 17, no. 4, pp. 283–291, 1976.
[257] P. A. Morris, "Decision analysis expert use," *Management Sci.*, vol. 20, no. 9, pp. 1233–1241, May 1974.
[258] M. S. S. Morton, *Management decision systems: Computer based support for decision making*. Cambridge, MA: Harvard Univ., 1971.
[259] M. Moskowitz, R. E. Schaefer, and K. Borcherding, "Irrationality of managerial judgments: Implications for information systems," *Omega*, vol. 4, no. 2, pp. 125–140, 1976.
[260] H. Moskowitz, "Regression models of behavior for managerial decision making," *The Int. J. of Management Sci.*, vol. 2, no. 5, pp. 677–690, 1974.
[261] J. Mumpower, V. Veirs, and K. R. Hammond, "Scientific information, social values and policy formulation," *IEEE Trans. Systems, Man., Cybern.*, vol. SMC-9, pp. 464–476, 1979.
[262] E. D. Neimark, "Current status of formal operations research," *Hum. Development*, vol. 22, pp. 60–67, 1979.
[263] R. E. Nisbett, and T. D. Wilson, "Telling more than we can know: Verbal reports on mental processes," *Psychological Rev.*, vol. 84, no. 3, pp. 231–259, May 1977.
[264] R. Nisbett and L. Ross, *Human Inference: Strategies and Shortcomings of Social Judgment*. Englewood Cliffs, NJ: Prentice-Hall, 1980.
[265] M. Nowakowska, "New Ideas in Decision Theory," *Int. J. Man–Machine System Studies*, vol. 11, pp. 213–234, 1979.
[266] P. C. Nutt, "Influence of decision styles on use of decision models," *Tech. Forecasting Social Change*, vol. 14, pp. 77–93, 1979.
[267] P. C. Nystrom and W. H. Starbuck, *Handbook of Organizational Design*, vols. I and II. New York: Oxford Univ., 1981.
[268] C. A. O'Reilly, III "The intentional distortion of information in organizational communication: A laboratory and field investigation, *Hum. Relations*, vol. 31, no. 2, pp. 173–193, 1978.
[269] C. W. Park, "A conflict resolution choice model," *J. Consumer Res.*, vol. 5, no. 2, pp. 24–137, Sept. 1978.
[270] C. W. Park, "A seven point scale and a decision maker's simplifying choice strategy: An operationalized satisficing plus model," *Organ. Behav. Hum. Perform.*, vol. 24, pp. 252–271, 1978.
[271] J. W. Payne, "Alternative approaches to decision making under risk: Moments versus risk dimensions," *Psychol. Bull.*, vol. 80, no. 6, pp. 439–453, 1973.
[272] ———, "Task complexity and contingent processing in decision making: An information search and protocol analysis," *Organ. Behav., Hum. Perform.*, vol. 16, pp. 366–387, 1976.
[273] J. W. Payne, M. L. Braunstein, and J. S. Carroll, "Exploring predecisional behavior: An alternative approach to decision research," *Organ. Behav. Hum. Perform.*, vol. 22, pp. 17–44, 1978.
[274] J. W. Payne, D. J. Laughhunn, and R. Crum, "Translation of gambler and aspiration level effects in risky choice behavior," *Management Sci.*, vol. 26, no. 10, pp. 1039–1060, Oct. 1980.
[275] J. W. Payne, "Information processing theory: Some concepts and methods applied to decision research," in *Cognitive Processes in Choice and Decision Behavior*, T. S. Wallsten, Ed. Hillsdale, NJ: Lawrence Erlbaum Ass., pp. 95–116, 1980.
[276] S. Penrod and R. Hastie, "Models of jury decision making: A critical review," *Psychol. Bull.*, vol. 86, no. 3, pp. 462–492, May 1979.

[277] W. D. Perreault and F. A. Russ, "Comparing multiattribute evaluation process models," *Behav. Sci.*, vol. 22, no. 6, pp. 423–431, Nov. 1977.

[278] G. F. Pitz, J. Heerboth, and N. J. Sachs, "Assessing the utility of multiattribute utility assessment," *Organ. Behav. Hum. Perform.*, vol. 26, pp. 65–80, 1980.

[279] C. R. Plott, "Axiomatic social choice theory: An overview and interpretation," *Am. J. Polit. Sci.*, vol. 20, pp. 511–596, 1976.

[280] C. R. Plott and M. E. Levine, "A model of agenda influence on committee decisions," *Amer. Econ. Rev.*, vol. 68, no. 1, pp. 146–160, Mar. 1978.

[281] M. I. Posner, *Cognition: An Introduction*. Glenview, IL: Scott Foresman, 1973.

[282] E. S. Quade, *Analysis for Public Decision*. New York: Elsevier, 1975.

[283] E. S. Quade and G. Majone, *Pitfalls of Analysis*. New York: Wiley, 1980.

[284] R. A. Radcliff, "A theory of memory retrieval," *Psych. Rev.*, vol. 85, pp. 59–108, 1978.

[285] H. Raiffa, *Decision Analysis*. Reading, MA: Addison-Wesley, 1968.

[286] D. W. Rajala and A. P. Sage, "On measures for decision model structuring," *Int. J. Syst. Sci.*, vol. 11, no. 1, pp. 17–31, 1980.

[287] _____, "On decision situation structural models," *Int. J. Policy Analysis, Inform. Syst.*, vol. 4, no. 1, pp. 53–81, 1980.

[288] R. H. Ranyard, "Elimination by aspects as a decision rule for risky choice," *Acta Psychol.*, vol. 40, pp. 299–310, 1976.

[289] R. H. Ranyard, "Risky decisions which violate transitivity and double cancellation," *Acta Psychol.*, vol. 41, no. 6, pp. 449–459, Oct. 1977.

[290] L. Rappoport and D. A. Summers, *Human Judgment in Social Interactions*. New York: Holt, Rhinehart, Winston, 1973.

[291] J. Rawls, *A Theory of Justice*. Cambridge, MA: Harvard University, 1971.

[292] D. Robey and W. Taggart, "Measuring managers' minds: The assessment of style in human information processing," *Academy of Management Rev.*, to be published.

[293] J. M. Robinson, *Decision Making in Urban Planning*. Beverly Hills, CA: Sage Pub., 1972.

[294] J. Rohrbaugh and P. Wehr, "Judgment analysis in policy formation: A new method for improving public participation," *Public Opin. Quar.*, vol. 42, no. 4, pp. 521–532, Winter 1978.

[295] J. Rohrbaugh, "Improving the quality of group judgment: Social judgment analysis and the Delphi technique," *Organ. Behav. Hum. Perform.*, vol. 24, pp. 73–92, 1979.

[296] M. Rokeach, *The Nature of Human Values*. New York: Free Press, 1973.

[297] E. Rosch and B. B. Lloyd, Eds. *Cognition and Categorization*. Hillsdale, NJ: Lawrence Erlbaum Ass., 1978.

[298] R. L. Rosenthal and B. J. Zimmerman, *Social Learning and Cognition*. New York: Academic Press, 1978.

[299] W. B. Rouse, *Systems Engineering Models of Human–Machine Interaction*. New York: North Holland, 1980.

[300] T. L. Saaty, *The analytic hierarchy process: Planning, priority setting, resource allocation*. New York, McGraw-Hill, 1980.

[301] A. P. Sage, *Systems engineering: Methodologies and application*. New York: IEEE, Wiley, 1977.

[302] _____, *Methodology for large scale systems*. New York, McGraw-Hill, 1977.

[303] A. P. Sage and D. W. Rajala, "On the role of structure in policy analysis and decision making," J. W. Sutherland, Ed., *Management Handbook for Public Administration*. New York: Van Nostrand, Reinhold, 1978, pp. 568–606.

[304] A. P. Sage and E. B. White, "Methodologies for risk and hazard assessment: A survey and status report," *IEEE Trans. Syst. Man, Cybern.*, vol. SMC-10, no. 8, pp. 425–446, Aug. 1980.

[305] A. P. Sage, "Desiderata for systems engineering education," *IEEE Trans. Syst. Man, Cybern.*, vol. 10, no. 12, pp. 777–780, Dec. 1980.

[306] _____, "Designs for optimal information filters," in P. C. Nystrom and W. H. Starbuck, Eds., *Handbook of Organizational Design*. New York: Oxford University, 1981, pp. 105–121.

[307] _____, "Systems engineering: Fundamental limits and future prospects," *IEEE Proc.*, vol. 69, no. 2, pp. 158–166, Feb. 1981.

[308] _____, "Methodological considerations in the design of large scale systems engineering processes," in *Large Scale Systems*, Y. Haimes, Ed. New York: North-Holland, 1981.

[309] _____, "A methodological framework for the design and evaluation of planning and decision support systems," *Comput. Elec. Eng.*, vol. 8, no. 2, pp. 87–102, June 1981.

[310] R. K. Sarin, A. Sicherman, and K. Nair, "Evaluating proposals using decision analysis," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-8, no. 2, pp. 128–131, Feb. 1978.

[311] G. B. Saunders and J. L. Stanton, "Personality as influencing factor in decision making," *Organ. Behav. Hum. Perform.*, vol. 21, pp. 241–257, Apr. 1978.

[312] E. H. Schein, *Organizational Psychology*. Englewood Cliffs, NJ: Prentice-Hall, 1972.

[313] W. Schneider and R. M. Shiffrin, "Controlled and automatic human information processing: Detection search and attention," *Psychol. Rev.*, vol. 84, pp. 1–66, 1977.

[314] P. J. Schoemaker and H. C. Kunreuther, "An experimental study of insurance decisions," *J. Risk, Insurance*, vol. 46, no. 4, pp. 603–618, 1979.

[315] P. J. Schoemaker, *Experiments on Decisions Under Risk, The Expected Utility Hypothesis*. Boston, MA: Martinus Nijhoff, 1980.

[316] L. P. Schrenk, "Aiding the decision maker—A decision process model," *Econometrics*, vol. 12, no. 4, pp. 543–557, 1969.

[317] P. R. Schulman, "Nonincremental policy making: Notes toward an alternative paradigm," *Amer. Polit. Sci. Rev.*, vol. 69, no. 4, pp. 1354–1370, Dec. 1975.

[318] R. L. Schultz and D. P. Slevin, *Implementing Operations Research/Management Science*. New York, Elsevier, 1975.

[319] F. P. Scioli, Jr. and T. J. Cook, *Methodologies for Analyzing Public Policies*. Lexington, MA: Lexington Books, 1975.

[320] G. Shafer, *A Mathematical Theory of Evidence*. Princeton, NJ: Princeton Univ., 1976.

[321] J. Shanteau and N. Anderson, "Integration theory applied to judgments of the value of information," *J. Exper. Psychol.*, vol. 92, no. 2, pp. 266–275, 1972.

[322] J. Shanteau, "Descriptive versus normative models of sequential inference judgment," *J. Exper. Psychol.*, vol. 93, no. 1, pp. 63–68, 1972.

[323] _____, "Component processes in risky decision making," *J. Exper. Psychol.*, vol. 103, no. 4, pp. 680–691, 1974.

[324] _____, "The concept of weight in judgment and decision making: A review and some unifying proposals," *Center Res. on Judgment, Policy*, Rep. 228, July 1980.

[325] D. A. Shelvugo, J. Jaccard, and J. Jacoby, "Preference, search, and choice: An integrative approach," *J. Consumer Res.*, vol. 16, pp. 168–176, Sept. 1979.

[326] J. M. Shepard and T. G. Hougland, Jr., "Contingency theory: 'Complexman' or 'complexorganization,'" *Academy of Management Rev.*, vol. 3, no. 3, pp. 413–427, July 1978.

[327] R. M. Shiffrin and W. Schneider, "Controlled and automatic human information processing: II Perceptual learning, automatic attending and a general theory," Psychol. Rev., vol. 84, pp. 127–190, 1977.

[328] S. M. Shugan, "The cost of thinking," *J. Consumer Res.*, vol. 7, pp. 99–111, 1980.

[329] C. R. Shumway, P. M. Maher, M. R. Baker, W. E. Sounder, A. H. Rubenstein, and A. R. Gallant, "Diffuse decision-making in hierarchical organizations: An empirical examination," *Management Sci.*, vol. 21, no. 6, pp. 697–707, Feb. 1975.

[330] R. A. Shweder, "Likeness and likelihood in everyday thought: Magical thinking in judgments about personality," *Curr. Anthropol.*, vol. 18, pp. 637–658, 1977.

[331] _____, "Rethinking culture and personality theory, a critical examination of two more classical postulates," *Ethos*, vol. 7, pp. 279–311, 1979.

[332] _____, Ed., *Fallible Judgment in Behavioral Research*. San Francisco, CA: Jossey Bass, 1980.

[333] R. S. Siegler, "Three aspects of cognitive development," *Cognitive Psychol.*, vol. 8, pp. 481–520, 1976.

[334] H. A. Simon, *Administrative Behavior*. New York, MacMillan, 1957.

[335] _____, "The functional equivalence of problem solving skills," *Cognitive Psychol.*, vol. 7, pp. 268–288, 1975.

[336] _____, "From substantive to procedural rationality," in *Method and Appraisal in Economics*, Spiro J. Latsis, Ed. New York: Cambridge Univ., pp. 129–148, 1976.

[337] H. Simon and S. Reed, "Modeling strategy shifts in a problem-solving task," *Cognitive Psychol.*, vol. 8, pp. 86–97, 1976.

[338] H. A. Simon, "Rationality as process and as product of thought," *Amer. Econ. Rev.*, vol. 68, no. 2, pp. 1–16, May 1978.

[339] ____, "On how to decide what to do," *Bell J., Econ.*, vol. 10, pp. 494–507, 1978.

[340] ____, *Models of Thought*. New Haven, CT: Yale University, 1979.

[341] ____, "Rational Decision Making in Business Organizations," *Amer. Econ. Rev.*, vol. 69, no. 4, pp. 493–513, Sept. 1979.

[342] ____, "Information processing models of cognition," *Ann. Rev. Psychol.*, vol. 30, pp. 363–396, 1979.

[343] ____, "The behavioral and social sciences," *Sci.*, vol. 209, pp. 72–78, July 1980.

[344] ____, "Cognitive science: The newest science of the artificial," *Cognitive Sci.*, vol. 4, pp. 33–46, 1980.

[345] P. Slovic and S. Lichtenstein, "Comparison of Bayesian and regression approaches to the study of information processing judgment," *Organ. Behav. Hum. Perform.*, vol. 6, pp. 649–744, 1971.

[346] P. Slovic, "Psychological study of human judgment: Implications for investment decision making," *J. Finance*, vol. 27, no. 4, pp. 779–799, Sept. 1972.

[347] P. Slovic, D. Fleissner, and W. S. Bauman, "Analyzing the use of information in investment decision making: A methodological proposal," *J. Bus.*, vol. 45, no. 2, pp. 283–301, 1972.

[348] P. Slovic, B. Fischhoff, and S. Lichtenstein, "Behavioral decision theory," *Ann. Rev. Psychol.*, vol. 28, pp. 1–39, 1977.

[349] E. R. Smith and F. D. Miller, "Limits of perception of cognitive processes: A reply to Nisbett and Wilson," *Psychol. Rev.*, vol. 85, pp. 355–363, 1978.

[350] H. T. Smith and T. R. G. Green, Eds., *Human interaction with computers*. New York: Academic Press, 1980.

[351] J. A. Sniezek, "Judgments of probabilistic events: Remembering the past and predicting the future," *J. Exper. Psychol.*, vol. 6, no. 4, pp. 695–706, 1980.

[352] M. Snyder and S. W. Uranowitz, "Reconstructing the past: Some cognitive consequences of person perception," *J. Personality and Social Psychol.*, vol. 36, pp. 941–950, 1978.

[353] P. O. Soelberg, "Unprogrammed decision making," *Sloan Management Rev.*, vol. 8, pp. 19–29, 1967.

[354] L. Solso, *Cognitive Psychology*. New York: Harcourt Brace Jovanovich, 1979.

[355] C. S. Spetzler and C. A. von Holstein, "Probability encoding in decision analysis," *Management Sci.*, vol. 22, no. 3, pp. 340–358, Nov. 1975.

[356] R. H. Sprague, Jr. and H. J. Watson, MIS concepts: Parts I and II, *J. of Systems Management*, vol. 26, no. 1, pp. 34–37, no. 2, pp. 35–40, 1975.

[357] W. H. Starbuck, "Organizations and their environments" in M. Dunnette, Ed., *Handbook of Industrial Psychology*. Chicago, IL: Rand McNally, 1976, pp. 1069–1123.

[358] M. K. Starr and M. Zeleny, *Multiple Criteria Decision Making*. New York: Elsevier, 1977.

[359] J. D. Steinbruner, *The Cybernetic Theory of Decision*. Princeton, NJ: Princeton University, 1974.

[360] R. J. Sternberg, "Component processes in analogical reasoning," *Psychological Review*, vol. 84, no. 4, pp. 353–378, 1977.

[361] E. Stokey and N. R. Zeckhauser, *A Primer for Policy Analysis*. New York: W. W. Norton, 1978.

[362] C. A. Stone and M. C. Day, "Competence and performance models and the characterization of formal operational skills," *Hum. Development*, vol. 28, pp. 323–353, 1980.

[363] J. W. Sutherland, Ed., *Management Handbook for Public Administrator*. New York: Van Nostrand, 1979.

[364] O. Svenson, "A unifying interpretation of different models for the integration of information when evaluating gambles," *Scand. J. Psychol.*, vol. 16, pp. 187–192, 1975.

[365] L. Svenson, "Process descriptions of decision making," *Organ. Behav. Hum. Perform.*, vol. 23, pp. 86–112, 1979.

[366] W. M. Taggart, Jr., *Information Systems: An Introduction to Computers in Organizations*. Boston, MA: Allyn and Bacon, 1980.

[367] W. M. Taggart, Jr. and M. O. Tharp, "A survey of information requirements analysis techniques," *Computing Surveys*, vol. 9, no. 4, pp. 273–290, 1977.

[368] R. N. Taylor and M. D. Dunnette, "Relative contribution of decision-maker attributes to decision processes," *Organ. Behav. Hum. Perform.*, vol. 12, pp. 286–298, 1974.

[369] R. N. Taylor, "Psychological determinants of bounded rationality: Implications for decision-making strategies," *Decision Sci.*, vol. 16, pp. 409–429, 1975.

[370] P. Terán, "Modes of behavior," *J. of Econ. Behav. and Organ.*, vol. 1, pp. 175–193, 1980.

[371] R. Thaler, "Toward a positive theory of consumer choice," *Econ. Behav. Organ.*, vol. 1, no. 1, pp. 39–60, Mar. 1980.

[372] W. Thorngate, "Efficient decision heuristics," *Behav. Sci.*, vol. 25, no. 3, pp. 219–225, May 1980.

[373] M. Tribus, "The decision process: A perspective," *Int. J. General Systems*, vol. 3, pp. 79–88, 1976.

[374] ____, "What Happens at the Moment of Decision? Meta Decisions, Emotions and Volitions," in *Human Decision Making*, L. Sjöberg, T. Tyszka, and J. A. Wise, Eds. vol. 2. Bodafors, Sweden: Doxa, 1980.

[375] ____, "Emotion and decision making," *Acta. Psychol.*, vol. 45, pp. 133–155, 1980.

[376] S. Toulmin, R. Rieke and A. Janik, *An Introduction to Reasoning*. New York: MacMillan, 1979.

[377] L. H. Tribe, "Policy sciences: Analysis or ideology?" *Philosophy and Public Affairs*, vol. 1, no. 1, pp. 66–101, 1971.

[378] L. H. Tribe and C. W. Schelling, *When Values Conflict*. Cambridge, MA: Ballinger, 1976.

[379] M. L. Tushman and D. A. Nadler, "Information processing as an integrating concept in organizational design," *Acad. of Management Rev.*, vol. 3, no. 3, pp. 613–624, July 1978.

[380] F. D. Tuggle and D. Gerwin, "An information processing model of organizational perception, strategy and choice," *Management Sci.*, vol. 26, no. 6, pp. 575–592, June 1980.

[381] A. Tversky, "Elimination by aspects: A theory of choice," *Psychol. Rev.*, vol. 79, no. 4, pp. 281–299, July 1972.

[382] ____, "Availability: A heuristic for judging frequency and probability," *Cognitive Psychol.*, vol. 5, pp. 207–232, 1973.

[383] A. Tversky and D. Kahneman, "Judgment under uncertainty: Heuristics and biases," *Science*, pp. 1123–1124, Sept. 27, 1974.

[384] A. Tversky and S. Sattath, "Preference Trees," *Psychol. Rev.*, vol. 86, no. 6, pp. 542–573, 1979.

[385] A. Tversky and D. Kahneman, "The framing of decisions and the psychology of choice," *Science*, vol. 211, pp. 453–458, Jan. 30, 1981.

[386] ____, *Evidential Impact of base rates*, Stanford Univ. Dep. of Psychol. Tech. Rep. 4, May 15, 1981.

[387] R. D. Tweney, M. E. Doherty, W. J. Worner, D. B. Pliske, C. R. Mynatt, K. A. Gross, and D. L. Arkkelin, "Strategies of rule discovery in an inference task," *Q. J. Exp. Psychol.*, vol. 32, pp. 109–123, 1980.

[388] A. H. Van De Ven and A. L. Delbecq, "The effectiveness of nominal, Delphi and interacting group decision making processes," *Acad. of Management J.*, vol. 17, no. 4, pp. 605–621, Dec. 1974.

[389] M. A. Vasarhelyi, "Man–machine planning systems: A cognitive style examination of interactive decision making," *J. Accounting Res.*, vol. 15, pp. 138–153, Spring 1977.

[390] A. Vinokur, "Review and theoretical analysis of the effects of group processes upon individual and group decisions involving risk," *Psychol. Bull.*, vol. 76, no. 4, pp. 231–241, Oct. 1971.

[391] A. Vinokur and E. Burnstein, "Depolarization of attitudes in groups," *J. of Personality Social Psychol.*, vol. 36, pp. 872–885, 1978.

[392] C. A. J. Vlek, "Coherence of human judgment in a limited probabilistic environment," *Organ. Behav. Hum. Perform.*, vol. 9, pp. 460–481, 1973.

[393] C. Vlek and P. Stallen, "Rational and personal aspects of risk," *Acta. Psychol.*, vol. 45, no. 1–3, pp. 273–300, Aug. 1980.

[394] V. H. Vroom and P. W. Yetton, *Leadership and Decision Making*. Pittsburgh, PA: Univ. Pittsburgh, 1973.

[395] T. S. Wallsten, "Processing information for decisions" in *Cognitive Theory*, vol. 2, N. J. Castellan, D. B. Pisoni, and G. Potts, Eds. Hillsdale, NJ: Lawrence Erlbaum Associates, 1977, pp. 87–116.

[396] ____, Ed., *Cognitive Processes in Choice and Decision Behavior*. Hillsdale, NJ: Lawrence Erlbaum Associates, 1980.

[397] J. N. Warfield, *Societal Systems: Planning Policy and Complexity*. New York: Wiley, 1976.

[398] D. A. Waterman and F. Hayes-Roth, *Pattern Directed Inference Systems*. New York: Academic, 1978.

[399] C. C. White, III and A. P. Sage, "A multiple objective optimization-based approach to choicemaking," *IEEE Trans. Syst. Man, Cybern.*, vol. SMC-10, no. 6, pp. 315–326, June 1980.

[400] A. Wildavsky, *Speaking Truth to Power: The Art and Craft of Policy Analysis*. Boston, MA: Little, Brown, 1979.

[401] D. Von Winterfeldt, "Structuring decision problems for decision analysis," *Acta Psychologica*, vol. 45, no. 1–3, pp. 71–93, 1980.

[402] P. Wright, "The harrassed decision maker: Time pressures, distractions, and the use of evidence," *J. Applied Psychol.*, vol. 59, no. 5, pp. 555–561, 1974.

[403] ———, "Self-insight into the cognitive processing of financial information," *Acct. Organ., Soc.*, vol. 2, no. 4, pp. 323–331, 1977.

[404] ———, "Citizen participation in public decisions: A comprehensive policy-capturing approach," in *Proc. 1979 Syst., Man, Cybern. Conf.*, Denver, CO, pp. 474–479, Oct. 1979.

[405] ———, "Properties of judgment models in a financial settings," *Organ. Behav., Hum. Perform.*, vol. 23, pp. 73–85, 1979.

[406] ———, "Cognitive information processing biases: Implications for producers and users of financial information," *Decision Sciences*, vol. 11, no. 2, pp. 284–298, Apr. 1980.

[407] J. F. Yates and C. M. Jagacinski, "Multiattribute evaluation reference effects: A reply to Barron and John," *Organ. Behav. Hum. Perform.*, vol. 25, pp. 375–383, 1980.

[408] J. F. Yates and L. G. Zukowski, "Characterization of ambiguity in decision making," *Behav. Sci.*, vol. 21, pp. 19–25, 1976.

[409] S. Zedeck, "An information processing model, an approach to the study of motivation," *Organ. Behav., Hum. Perform.*, vol. 18, no. 1, pp. 47–77, Feb. 1977.

[410] E. Zaidel, "The elusive right hemisphere of the brain," *Eng. Sci.*, vol. 42, no. 1, pp. 10–12, 1978.

[411] R. B. Zajonc, "Feeling and thinking-preferences need no inferences," *Amer. Psychol.*, vol. 35, no. 2, pp. 151–175, Feb. 1980.

[412] D. E. Zand and R. E. Sorensen, "Theory of change and the effective use of management science," *Admin. Sci. Quart.*, vol. 20, no. 4, pp. 532–545, Dec. 1975.

[413] R. W. Zmud, "On the validity of the analytic-heuristic instrument utilized in the Minnesota experiments," *Management Sci.*, vol. 24, pp. 1088–1090, 1978.

[414] R. W. Zmud, "Perceptions of cognitive style: Acquisition, exhibition, and implications for information system design," *J. Management*, vol. 5, no. 1, pp. 7–20, 1979.

[415] R. W. Zmud, "Individual differences and MIS success: A review of the empirical literature," *Management Sci.*, vol. 25, no. 10, pp. 966–979, Oct. 1979.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

MIS IS A MIRAGE

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984

Harvard Business Review
January-February 1972.

John Dearden

MIS is a mirage

Can a single, integrated system be devised to fill all of management's information needs? Only if Superman lends a helping hand

Foreword

Every company of any size has many information systems, both formal and informal. The formal systems it uses cover such a variety of territory that one man cannot possibly comprehend the mass of details and principles required to design a single supersystem that embraces them all. Even a group of systems experts cannot create such a supersystem, the author argues, because the components that must be amalgamated are too different in their natures to be fused together effectively. After demonstrating the futility of the MIS approach, the author recommends practical steps for reforming defective information systems.

Mr. Dearden is Professor of Business Administration at the Harvard Business School. He is well known to HBR readers, especially for his significant contributions to the theory of divisional control. [A list of Mr. Dearden's previous HBR articles on information systems appears in the ruled insert on page 98.]

Some years ago I expressed the opinion that "of all the ridiculous things that have been foisted on the long-suffering executive in the name of science and progress, the real-time management information system is the silliest." [1]

I no longer believe this statement is true. We now have something even sillier: the current fad for "the management information system," whether it is called the Total System, the Total Management Information System, the Management Information System, or simply MIS.

I certainly do not mean to suggest that a company does not need good management information systems—nothing could be further from the truth. But the notion that a company can and ought to have an expert (or a group of experts) create for it a single, completely integrated supersystem—an "MIS"—to help it govern every aspect of its activity is absurd.

For many businessmen, it is probably inconceivable that the lofty phrases and glittering promises surrounding the MIS conceal a completely unworkable concept. Yet this is exactly what I propose to demonstrate—that a company that pursues an MIS embarks on a wild-goose chase, a search for a will-o'-the-wisp.

Let me first try to explain what I understand by the "MIS concept" and examine its alleged advantages, and then show why the concept is unworkable. Then I shall be in a position to recommend some practical remedies for defective management information systems, which certainly constitute a real problem for executives today.

Confusion between terms

It is difficult even to describe the MIS in a satisfactory way, because this conceptual entity is embedded in a mish-mash of fuzzy thinking and incomprehensible jargon. It is nearly impossible to obtain any agreement on how MIS problems are to be analyzed, what shape their solutions might take, or how these solutions are to be

1. "Myth of Real-Time Management Information," HBR May-June 1966, p. 123.

implemented. This confusion makes it very difficult to attack the concept, because no matter what assumptions a critic makes about the nature of the MIS approach, a proponent can always reply that *his* use of the term is different from others'.

But there is a common thread which runs through the various uses of the term, a thread that at once unifies but also subverts the MIS literature. This thread is the computer-based information system.

Computer-based activity . . .

Wherever the MIS is discussed, it is almost invariably stated that a management information system does not necessarily require a computer and that many forms of management information are not computer-based.

Yet, if one looks at what is actually being discussed, he quickly discovers that the term "MIS" is used, essentially, to stand for "computer-based information systems." For example, a recent article in *Business Week* read as follows:

"[Some], concerned that systems analysts are . . . a 'mixed bag' whose training and knowledge are a hit-or-miss proposition, are convinced that management information systems [MIS] is the emerging field in business administration. Both Wharton and MIT have tailored programs especially for systems specialists, but no school has gone further than the University of Minnesota, whose B-school now offers MS and PhD degrees in management information systems and has launched an MIS research center. Since the center's opening three years ago, MIS Director Gordon B. Davis and his staff have worked to develop 12 new systems-related courses—from on-line, real-time systems to a seminar on software. In addition, the program's 50 MS and 22 PhD candidates spend a good portion of their time alone and in teams at work on actual computer problems in industry." [2]

It seems evident to me that MIS education as described here is principally education in computer-based information systems.

It is vital to note, first of all, that the information generated by this kind of system does not include a great deal of the information that is most important to management—especially, important *qualitative* information. Second, a specialist group that develops such a system is usually responsible for implementing only one part of any of a company's individual management information systems—namely, that part that interfaces directly with the computer. For example, such a group has little (if anything) to do with specifying the nature of an accounting and financial control system, although it may be responsible for the computer programming this system employs.

My conclusion, therefore, is that such a group has little impact on most of the information supplied to management, particularly at upper levels. Consequently it is ridiculous to say that it creates [or *can* create] a total management information system.

. . . vs. MIS

To the extent that MIS refers only to company information systems that use a computer base and to the extent that everyone understands this limitation, I have no serious quarrel with the trend to MIS; it is vital that management tightly control its computer-based information systems, and in general the so-called MIS groups seem designed to guarantee a tight rein to management.

In my experience, however, such a limited definition of MIS is *not* what advocates of this approach to information systems mean when they use the term. They intend something novel and far more global, some entity that can provide revolutionary benefits we cannot derive from the traditional approach. Walter Kenneron suggests this definition of the MIS:

"A management information system is an organized method of providing past, present and projection information relating to internal operations and external intelligence. It supports the planning, control and operational function of an organization by furnishing uniform information in the proper time-frame to assist the decision-maker." [3]

This is approximately what I perceive most people to mean by MIS. And if this definition seems grandiose, I can only remark that "*the* management information system" describes a grandiose idea. If the definition were less global in its scope, it would not measure up to the term. If, for example, one were to limit the definition to the context of a company's financial accounting programs, he would have to speak of the *financial* MIS of the company, rather than its general MIS.

2 June 1, 1971, p. 56.

3 "MIS Overview," Data Management, September 1970.

However, in practice, no such limitations are intended. Kenneron's inclusive definition of the MIS approach is quite consistent with the nearly universal benefits claimed for it.

The MIS approach

Given this inclusive definition, how is management to apply it? In other words, how should management think about the problem of setting up an MIS?

Fundamental assumptions

First, it appears that if management wishes to subscribe to the theory of the MIS, it must make up its mind to accept two fundamental (if highly questionable) assumptions that are quite different from traditional ones made in this area:

1. Management information is a subject for study and specialization. That is, it is sufficiently homogeneous so that a set of principles and practices can be esablished for evaluating all management's information needs and satisfying them. In short, the MIS approach attacks all the problems of management information as a whole, rather than by individual areas, such as finance and marketing. This homogeneity is a necessary assumption, since without it there is no reason why general solutions to a management's information requirements can be found.

2. The systems approach can and should be used in analyzing management's information requirements. Proponents claim the systems approach is necessary for mastering the sprawl of requirements and for synthesizing the general MIS solution. (I shall have more to say about the systems approach later.)

Diagnosis & development

Once management has accepted these two assumptions, it can begin to develop an MIS program. As the theory goes, there seem to be two techniques for setting to work:

☐ Management can hire an MIS expert to act as a superconsultant to the president of the company. This expert studies the types of problems that the president must solve, the decisions that he must make, and so forth, and recommends methods for satisfying the president's total information requirements. He then drops to lower levels of management and provides the same services there.

In general, the expert depends on others to implement his recommendations. For example, the controller becomes responsible for changing the cost accounting system in the way the consultant recommends.

☐ Management can create a staff department that reports to the top. This group is responsible for the company's computer-based systems but also provides the same type of diagnoses and evaluations as the superconsultant.

The staff group, unlike the consultant, usually has responsibility for implementation.

Its alleged advantages . . .

Under this approach, then, either a single person or a group of persons is responsible for developing and overseeing the construction of the entire management information system. This concentration of authority and responsibility in the hands of systems experts supposedly creates a number of significant advantages:

☐ Experts schooled in the MIS "discipline" can analyze management's information needs more effectively than can the people traditionally responsible for satisfying them. Moreover, these experts can better determine which techniques will best meet these needs.

☐ Because the MIS is developed as a unified, single system, rather than as a number of separate systems, it is completely coordinated and completely consistent.

☐ Information needs are determined from the top down. Hence the top will be in better control; the frequent practice of letting lower management decide what information will pass upward is eliminated.

☐ The company reduces its direct information costs by eliminating systems. Also, the MIS itself is cheaper to run because it has been designed by information experts who know the most economical means for satisfying management's information needs.

☐ Since one expert or group is responsible for the system, management's desire that the system be kept up-to-date can readily be satisfied.

In short, the proponents promise, experts can design an MIS that is more effective, more efficient, more consistent, and more dynamic than the haphazard aggregate of individual systems a company would otherwise employ.

These are impressive advantages that any manager would enjoy, and doubtless this ap-

proach was developed to solve the real problems of poor information that have been plaguing management with increasing frequency. The growing complexity and the pace of change of modern business, especially in the last ten years, have surely made many information systems obsolete and many more inadequate for present tasks.

Equally, the last ten years have seen the extensive development of information technology, management science, and systems analysis—a development that has been accompanied by rapid growth in the number of experts working in information systems.

To some—that is, the proponents of MIS—it seemed logical to centralize the development and control of information systems in the hands of these experts. After all, the problems that beset information systems have been the result of change and growth, they reasoned; and these problems could perhaps be solved by using the new information technology that had been developing simultaneously.

Several companies have tried this approach, and many people currently advocate it. In spite of its apparent logic, however, I know of no company in which it has worked out. This fails to surprise me because, as I have already implied, I believe the whole MIS approach is fundamentally fallacious.

. . . & its real fallacies

There are four fallacies and one serious misconception inherent in the MIS approach, as I have described it. The fallacies are these:

◊ Management information is sufficiently homogeneous so that it can be made an area of specialization for an expert.

◊ If the different information systems ordinarily used by a company are developed separately, the resulting management information system will necessarily be uncoordinated and therefore inefficient and unsatisfactory.

◊ The "systems" approach is a new boon to business administration.

◊ It is practicable to centralize the control over a company's entire management information system.

The misconception is this:

◊ The specialist expertise that creates a good logistics system for a company can extend its talents into the broad domain of general company activity and create a general management information system.

There is no reason to suppose an MIS group can actually do this—in fact, there is good reason to think it cannot.

Let me refute these errors one by one.

1. The true MIS expert does not and cannot exist.

A complete management information system consists of such a huge assortment of different types of activities that no man can possess a broad enough set of special skills to apply to even a small proportion of them. Consider the skills required to build any one of these individual information systems.

The financial accounting and control system: This includes preparation of financial statements, development of budgets and long-range plans, analyses of capital investments, publication of product costs, and so forth.

Traditionally, the controller is responsible for all these financial subsystems; with respect to the financial information systems, he plays the role that the MIS expert is supposed to play in the general management information systems. In complementary fashion, the MIS expert must have a thorough understanding of the controller's systems function.

The logistics information system: This system controls the flow of goods from the purchase of raw materials to the physical distribution of the finished products. Next to the financial control system, it is probably the most comprehensive information system in the typical manufacturing business.

A logistics system normally consists of several subsystems of varying degrees of independence. For example, there could be distinct systems for different product lines. Within each product line, furthermore, there could be subsystems for procurement, production scheduling, finished goods, inventory control, and so forth, and still others for plant utilization and expansion. Depending on its industry, a company has a larger or smaller number of complex, interrelated logistics information subsystems.

The critical point to note here is that the logistics information system is almost completely different from the financial information system. In point of fact, most of the skills needed to develop financial information systems are of

no use in developing logistics information systems and vice versa. Even the user relationships are different. In building a financial information system, the controller develops a system that provides information for management outside the finance function, whereas logistics information is normally developed and used by the people directly concerned with logistics.

Furthermore, logistics subsystems frequently have little in common with each other, so that an expert in one type of subsystem might not be able to transfer his expertise to a different type. For example, there may be little or no similarity between a procurement information system and a finished-goods distribution system. Like the financial system, the logistics information system or subsystem is a job for a specialist.

The marketing information system: Like the two systems just described, the marketing information system can also consist of a number of subsystems. A company may maintain separate subsystems for separate product lines; and within a product line, it may maintain further subsystems for advertising and sales promotion, short-term sales forecasting, long-term sales forecasting, product planning, and so forth.

Again, the critical point is this—a marketing information system is almost completely different from the other two systems. Consequently, expertise in either or both of the other systems would be of limited value in developing a marketing information system and vice versa.

Legal services, industrial relations, and public relations: One of the major purposes of each of these staff functions is to provide top-management with specialized information different from that provided by any other staff office and different from that provided by the three information systems previously described.

R&D reporting: The information system management requires in this area is distinct from all others, and expertise in these other areas offers limited help in developing an R&D information system.

In short, except in the small company (which probably needs only simple systems), there are several information systems that have very few similarities and many wide differences. Consequently, it makes no sense to regard the processes of developing and implementing these several management information systems as constituting a single and homogeneous activity.

I conclude that few, if any, individuals have the training to call themselves experts in management information systems. Indeed I believe it is much more practical to teach the new information technology to the functional experts than to teach information technologists functional specialties. After all, the man who could master all the functional specialties—the true MIS expert—would have to be an intellectual superman; and hence he does not and cannot exist except, perhaps, as a very rare exception.

If an MIS can be implemented at all, it can only be implemented by a staff group, and one of considerable size.

2. Coordinated systems for functional areas can be developed without a 'total systems approach.'

"Unless you develop the MIS as a single, integrated system, all you will get is a bunch of unrelated, uncoordinated, ineffective systems." If I have heard this statement once, I have heard it a hundred times; and it still is not true.

I have seen many systems that have intricate interfaces with one another and that are still efficient and effective. In the automobile industry, for example, the development of a new model car involves many functions—styling, engineering, product planning, finance, facility planning, procurement, and production scheduling. Each functional unit develops its internal information system for controlling its part of the operation; in addition, at each interface, the functional units exchange the information necessary to maintain coordination between them.

If an information system is ineffective, the cause is very likely to be the incompetence of the people responsible for it, not the absence of the general MIS approach. In this connection I might quote William M. Zani:

"Most companies have not conceived and planned their management information system with any significant amount of attention to their intended function of supporting the manager as he makes his decisions."[4]

Zani goes on to suggest a new approach to developing an MIS as a solution to this situation. My solution would be to make some personnel changes, because anyone who fails to design an information system for its users is incompetent.

Such incompetence is very prevalent. I have seen dozens of companies where management is

4. "Blueprint for MIS," HBR November-December 1970, p. 95.

not receiving half the relevant accounting information that could be made available if the financial information system had been properly designed in the first place. And although I am not sufficiently expert in other types of information systems to know whether the same situation exists there, I have no reason to believe accounting is worse than the others.

To assert that such problems as these result from the independent development of different information systems, rather than from sheer and ordinary incompetence, is simply ridiculous—and to recommend the "MIS cure" is even more ridiculous. To ensure that a company has efficient information systems which are well coordinated with one another, management need only bear down on the personnel in the various functional areas who are responsible.

3. 'The systems approach' is merely an elaborate phrase for 'good management.'

There are many definitions of the systems approach, but the following is representative:

"The systems approach to management is basically a way of thinking. The organization is viewed as an integrated complex of interdependent parts which are capable of sensitive and accurate interaction among themselves and with their environment." [5]

What does this mean? It took me some time to figure it out.

When the systems approach first appeared in the literature, I had a great deal of difficulty understanding the concept; and my confusion increased until I started asking people this question: "What would an executive do differently if he were to adopt the systems approach in place of the traditional one?"

Without exception, the replies I received made assumptions about the traditional approach that simply are not valid. For example, some assumed that the executive perceives his organization as static; others, that he fails to consider the interaction of related variables. In other words, the replies were predicated on an incompetent, even a stupid, executive.

Thus I concluded that the alleged advantages of the systems approach really result from the difference between an adequate and an inadequate manager. If you doubt this, I invite you to ask the question I did the next time you hear someone champion the systems approach to management.

It is therefore not surprising that good managers follow the systems approach, because this approach is merely the ancient art of management. Would a competent business executive plan a major expansion program without considering the sources and timing of funds, the availability of people, the possible reactions of competitors, and so forth? Certainly not. And he would consider them in relation to one another.

My conclusion, then, is that the systems approach is precisely what every good manager has been using for centuries. The systems approach may be new to science and to weapons acquisition, but it is certainly not new to business administration.

At this point, let me summarize briefly. First, an MIS would have to be developed by a *group* composed of experts in the various types of information systems used by management. This must be so because the possibility that a single individual will be expert in *all* types of information is remote. Second, the approach taken by the MIS group would be approximately the same as that taken by any competent and expert manager working in one of the functional information systems.

How, then, does the MIS approach differ from the traditional approach to information systems?

The only difference I can see is that a company's management information system would be the responsibility of one centralized group; whereas, traditionally, the information systems experts have been located in the various functional areas. This brings me to the last fallacy—that such centralization is practicable.

4. Centralizing the control of a company's information systems in a staff group creates problems that are insoluble; therefore it is simply not feasible.

It is theoretically possible to assemble a staff MIS group that is sufficiently large and diversified to have expertise in all the formal information systems described earlier—marketing, manufacturing [logistics], finance, and so forth. But to organize this group properly, the company should appoint an executive vice president for information to supervise the work of the group—that is to say, the systems of the staff vice presidents, the controller, the logistics information group, the marketing information group, and so

5. Spyros Makridakis, "The Whys and Wherefores of the Systems Approach," *European Business*, Summer 1971.

forth. But what would this accomplish? Let me ignore the fact that no sane manufacturing or marketing executive would delegate the responsibility for his information system.

One result might be that this executive vice president for information would promote better coordination between functional areas. On the other hand, of course, the problems of coordination would drastically increase in the manufacturing and marketing areas because the responsibility for the information systems had been separated from the people who hold the line responsibility. And in any event, simply having all of the information groups, including the MIS group, report to a single executive would hardly change the *approach* to developing information systems. Thus the special value of the MIS approach is still obscure.

In short, it seems to me that if any of the MIS people are competent to tell the functional experts what to do, they should be in the functional area. I see no logical way to centralize the responsibility for all the management information systems.

Significant misconception

If the MIS approach is as fallacious as I believe it to be, how has it been able to maintain even a superficial credibility?

The answer, as I have hinted earlier, is this: the early success of information technology in renovating logistics systems has been so great that there is a natural inclination to try the same methods on the company information systems as a whole.

This misconception has evolved in a natural enough way. Responsibility for a logistics system has traditionally been divided among several executives—e.g., in purchasing, in manufacturing, and in marketing. This divided responsibility has often resulted in poor coordination throughout the system. Furthermore, the people responsible for the system have often been old-fashioned in their methods and relatively unskilled in information techniques. Thus a vacuum has frequently existed with respect to the responsibility for a company's logistics information system into which the burgeoning information technology has moved easily and successfully.

However, as we have seen, there is no reason to suppose that the principles of information technology used so successfully in the logistics area can be generalized to apply to the other management information systems within a company or to the management information system considered as a whole.

Thus, when a group of experts has completed its overhaul of the logistics system, it will not be in a position to attack the financial, marketing, or any other system. First, the group will not have the specialist expertise required. Second, the type of problems the group may have found in the logistics area will almost certainly not exist in other areas if the staffs in these other areas are competent. Third, there will be no responsibility vacuum as in the logistics area; the MIS group will not be in a position to take over by default.

If you have any doubt about the validity of these statements, I suggest that you examine the kinds of things that any MIS group is doing. Outside of the routine computer systems, you will almost certainly find them concerned basically with parts of the *logistics* information system only.

Roots of poor information

So far this article has been quite negative. Now I should like to suggest some positive actions to mitigate the information crisis, if it can be called that. Before I propose these actions, however, it is appropriate to review the causes of management information problems.

As I have pointed out, the principal cause of poor information systems is that we have put incompetent or ineffective people in charge of these systems.

The secondary causes are somewhat more complicated.

Growing use of computers

Computers and computer-related systems activities have been growing very rapidly, and currently the cost of these activities has become very significant in many companies. In spite of large expenditures, however, the quality of the information available to management appears unimproved.

One reason is, of course, that some computer installations are not run effectively. Another is that the computer-based information systems have been oversold; management has been led to expect much more than it has received. In other words, management's dissatisfaction with its information occurs, not from any deteriora-

tion in its information systems, but from its inflated expectations.

Interface conditions

Individual systems change and improve at different rates, and this creates problems at the interfaces between them. For example, operations research techniques, used in modern logistics systems, require much more sophisticated cost accounting information than traditional cost accounting techniques can generate. Problems can also occur at the interface between production and marketing, because production-scheduling techniques are frequently much more sophisticated than the techniques ordinarily used in market forecasting.

In general, the benefits of advanced techniques may be largely lost where they are dependent on primitive ones. (To some extent, of course, the problem of proper coordination at the interfaces reflects the competency of the staff involved. Other things being equal, only an incompetent would use an advanced technique whose effectiveness would be undermined by inadequate support.)

Rapidity of change

Many companies are changing very rapidly, and it is necessary that their information systems keep pace. In some companies, information systems are not keeping pace. To some extent, this is caused by the inability of the staff personnel traditionally responsible for information systems to react to change. After all, many people who were once perfectly adequate in a relatively static situation become ineffective in a dynamic situation.

Greater management challenge

Management must always operate with insufficient information. And frequently, the more important the decision, the greater the uncertainty. In many areas the truth of these statements is becoming more salient because, while the role of management is becoming more complex, the new information technology is not helping significantly.

For example, I have spent many years working on control systems for decentralized companies. The problems of control in such companies today are much more difficult than they were ten years ago—increases in size, complexity, and geographical dispersion have made control much more difficult. Yet the new information technology has been of little help in this area, simply because the problems of controlling decentralized divisions do not lend themselves to computerized or mathematical solutions.

Accordingly, it is important to realize that part of our information crisis results from the nature of the present business environment. We shall simply have to live with it. This does not mean, of course, that we should not continue trying to improve the situation.

Toward real solutions

Any company that believes it is facing genuine management information problems and wants to solve them should consider the following measures.

1. Place competent people in each of the formal information systems.

To my mind there is no question that incompetency is the leading cause of problems in many management information systems. Hence the obvious answer is to retrain or replace the incompetents.

2. Examine the interfaces.

This is best done in connection with system evaluation, and the examination should focus on these evaluative questions:

O Is there adequate communication between individual groups at all important interfaces?

The executive might bear in mind formal techniques such as scheduled meetings and formal agreements.

O Does each group involved in an interface know enough about the other interfacing systems to do its job effectively?

This is a question of education. For example, cost accountants should know enough about company operations-research models to be sure these models are providing correct information; or, at the very least, they should be able to explain to the OR group the relevant limitations of the information their group can supply. On the other hand, the OR people should know enough about cost accounting to ask for the right type of data and to appreciate the limitations in the data they receive.

But although this is principally a matter of

education, it may well be that some staff members are not intellectually capable of handling interface requirements, and they may have to be replaced.

3. Examine the logistics system.

Originally many logistics systems were organized for manual data processing and have never been changed. Equally, the procurement, production, and distribution functions typically report to different executives, and consequently no one is formally responsible for the logistics information system. Since it is here that computers and information technology are most applicable, management should evaluate its logistics area and, where appropriate, reorganize it and make a staff unit, responsible for its logistics information system, report to the company officer who directs the logistic system itself.

Readers particularly interested in this topic may wish to consult these previous HBR articles by Professor Dearden:

"Can Management Information Be Automated?" March-April 1964, p. 128.

"Computers: No Impact on Divisional Control," January-February 1967, p. 99.

"How to Organize Information Systems," March-April 1965, p. 65.

"Myth of Real-Time Management Information," May-June 1966, p. 123.

For more perspective on the CAIS-MIS controversy, readers may also find these HBR articles helpful:

Warren F. McFarlan, "Problems in Planning the Information System," March-April 1971, p. 75.

William M. Zani, "Blueprint for MIS," November-December 1970, p. 95.

4. Organize a central computer group for systems control.

Computer use will continue to expand, and it is vital that management maintain central control over computers and computer-based information systems.[6] Such a group should be responsible for overseeing all computer-related work—for long-range planning, coordination, and control of all computer acquisitions and applications. In addition, it should be responsible for coordinating computer-based systems and might even undertake the systems and implementation work in a situation where several organization groups use the same data base.

Most companies already have such groups. Some are even called "MIS groups," although, in reality, they have authority only over computer-related work.

5. Create an administration vice president, if one does not already exist.

I recommend the creation of an office to which the following report:

- ○ The controller.
- ○ The treasurer.
- ○ The computer and systems group.
- ○ The legal office.
- ○ The industrial relations office.
- ○ Other offices for company relations (that is, public and governmental).
- ○ Organization planning.

The marketing, manufacturing, and R&D groups would continue to be independent.

Such an office has several advantages:

☐ It provides better control over the staff activities. The increasing number of staff operations, together with their increasing specialization, has made it nearly impossible for the president to exercise real control here. An administrative vice president can exercise much more effective control over the size and direction of these activities.

☐ It provides a practical alternative to locating the computer and systems group in the controller's office. An administrative vice president can provide effective supervision and, at the same time, maintain an objectivity that a controller often finds difficult because of his involvement with specific computer applications.

☐ It allows the company to handle miscellaneous projects easily—for example, an evaluation of a functional information system or an analysis of the formal information entering the president's office. To take care of nonrecurring or particularly pressing information systems problems, frequently the best arrangement is to organize temporary task forces that report to the administrative vice president.

☐ It simplifies the process of coordinating staff offices.

However, I would not make the administrative vice president or the offices reporting to him

6. See Warren F. McFarlan, "Problems in Planning the Information System," HBR March-April 1971, p. 75.

responsible for the *entire* management information system. Marketing, manufacturing, and R&D would all be responsible for their own information systems. Also, the different activities reporting to his office would develop their information systems in relative independence, except where interface communications are in question.

Questions for my critics

Inevitably, I shall be accused of setting up a straw issue in this article and then demolishing it.

If the MIS approach really embraces only computer-based information systems or centralized logistics systems, then I *have* set up a straw issue. No harm has been done, however, because I have at least clarified the meaning of "MIS."

But I cannot believe the concept is meant to embrace only this. I have done my best to discover what the MIS approach really is, through talking with its proponents and studying its literature; and this article honestly represents my best understanding.

If I am correct in believing that the approach pretends to embrace more than computerized systems and logistics, then I have not set up a straw issue. And those who doubt my conclusions, negative as these may be, would be wise to ask themselves the following questions before they take up the pen of protest:

- Which information systems are to be included in the MIS?
- What kinds of experts are to be included in an MIS group, and what training shall they have?
- Where is this group to fit into the corporate organization? In particular, what will happen to the staff groups from the controller's office, the legal department, the marketing research department, and so forth?
- What authority is the MIS group to have? Is it to have authority to design and implement systems, or is it to serve in an advisory function only?
- What can this group accomplish that cannot be better accomplished by placing information specialists under functional groups?

Arguing the viability of the MIS approach is pointless unless answers to these questions are set forth clearly. And the clearer the answers, I believe, the more transparent the MIS mirage.

Scientists & critics

The criticism of science in the twentieth century is a kind of *lèse majesté* somewhat equivalent to criticizing the Roman Catholic Church in the twelfth century. It is seldom realized that every form of intellectual endeavor with the exception of science has both practitioners and informed critics. These critics in the fields of poetry, fiction, drama, painting, sculpture, music, etc., not only function as interpreters of the practitioners to the general public, but as critics who compare and evaluate the work of the practitioners. A critic who never wrote a poem, composed a score, or painted a picture may perform the valuable service of noting that a particular poem, score, or painting is uninspired, shoddy, or imitative. Furthermore, no one seriously supposes that the work of such critics constitutes censorship or restriction on the free creative spirits of poets, musicians, or painters. But scientists have insisted that any criticism of their work does constitute censorship or a failure to appreciate the necessity for "basic" [which sometimes should be read as trivial or useless] research. . . .

The way things are now, if someone can get by with preempting the use of the term "science" he is relatively free from exposure as a charlatan even if he is one.

Mortimer Taube,
Computers and Common Sense,
New York, Columbia University
Press, 1961, pp. 121, 124.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

MANAGEMENT MISINFORMATION SYSTEMS

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984

MANAGEMENT SCIENCE
Vol. 14, No. 4, December, 1967
Printed in U.S.A.

6

MANAGEMENT MISINFORMATION SYSTEMS*

RUSSELL L. ACKOFF

University of Pennsylvania

Five assumptions commonly made by designers of management information systems are identified. It is argued that these are not justified in many (if not most) cases and hence lead to major deficiencies in the resulting systems. These assumptions are: (1) the critical deficiency under which most managers operate is the lack of relevant information, (2) the manager needs the information he wants, (3) if a manager has the information he needs his decision making will improve, (4) better communication between managers improves organizational performance, and (5) a manager does not have to understand how his information system works, only how to use it. To overcome these assumptions and the deficiencies which result from them, a management information system should be imbedded in a management control system. A procedure for designing such a system is proposed and an example is given of the type of control system which it produces.

The growing preoccupation of operations researchers and management scientists with Management Information Systems (MIS's) is apparent. In fact, for some the design of such systems has almost become synonymous with operations research or management science. Enthusiasm for such systems is understandable: it involves the researcher in a romantic relationship with the most glamorous instrument of our time, the computer. Such enthusiasm is understandable but, nevertheless, some of the excesses to which it has led are not excusable.

Contrary to the impression produced by the growing literature, few computerized management information systems have been put into operation. Of those I've seen that have been implemented, most have not matched expectations and some have been outright failures. I believe that these near- and far-misses could have been avoided if certain false (and usually implicit) assumptions on which many such systems have been erected had not been made.

There seem to be five common and erroneous assumptions underlying the design of most MIS's, each of which I will consider. After doing so I will outline an MIS design procedure which avoids these assumptions.

Give Them More

Most MIS's are designed on the assumption that the critical deficiency under which most managers operate is the *lack of relevant information*. I do not deny that most managers lack a good deal of information that they should have, but I do deny that this is the most important informational deficiency from which they suffer. It seems to me that they suffer more from an *over abundance of irrelevant information*.

* Received June 1967.

This is not a play on words. The consequences of changing the emphasis of an MIS from supplying relevant information to eliminating irrelevant information is considerable. If one is preoccupied with supplying relevant information, attention is almost exclusively given to the generation, storage, and retrieval of information: hence emphasis is placed on constructing data banks, coding, indexing, updating files, access languages, and so on. The ideal which has emerged from this orientation is an infinite pool of data into which a manager can reach to pull out any information he wants. If, on the other hand, one sees the manager's information problem primarily, but not exclusively, as one that arises out of an overabundance of irrelevant information, most of which was not asked for, then the two most important functions of an information system become *filtration* (or evaluation) and *condensation*. The literature on MIS's seldom refers to these functions let alone considers how to carry them out.

My experience indicates that most managers receive much more data (if not information) than they can possibly absorb even if they spend all of their time trying to do so. Hence they already suffer from an information overload. They must spend a great deal of time separating the relevant from the irrelevant and searching for the kernels in the relevant documents. For example, I have found that I receive an average of forty-three hours of unsolicited reading material each week. The solicited material is usually half again this amount.

I have seen a daily stock status report that consists of approximately six hundred pages of computer print-out. The report is circulated daily across managers' desks. I've also seen requests for major capital expenditures that come in book size, several of which are distributed to managers each week. It is not uncommon for many managers to receive an average of one journal a day or more. One could go on and on.

Unless the information overload to which managers are subjected is reduced, any additional information made available by an MIS cannot be expected to be used effectively.

Even relevant documents have too much redundancy. Most documents can be considerably condensed without loss of content. My point here is best made, perhaps, by describing briefly an experiment that a few of my colleagues and I conducted on the OR literature several years ago. By using a panel of well-known experts we identified four OR articles that all members of the panel considered to be "above average," and four articles that were considered to be "below average." The authors of the eight articles were asked to prepare "objective" examinations (duration thirty minutes) plus answers for graduate students who were to be assigned the articles for reading. (The authors were not informed about the experiment.) Then several experienced writers were asked to reduce each article to ⅓ and ½ of its original length only by eliminating words. They also prepared a brief abstract of each article. Those who did the condensing did not see the examinations to be given to the students.

A group of graduate students who had not previously read the articles were then selected. Each one was given four articles *randomly* selected, each of which

on in one of its four versions: 100%, 67%, 33%, or abstract. Each version of each article was read by two students. All were given the same examinations. The average scores on the examinations were then compared.

For the above-average articles there was no significant difference between average test scores for the 100%, 67%, and 33% versions, but there was a significant decrease in average test scores for those who had read only the abstract. For the below-average articles there was no difference in average test scores among those who had read the 100%, 67%, and 33% versions, but there was a significant increase in average test scores of those who had read only the abstract.

The sample used was obviously too small for general conclusions but the results strongly indicate the extent to which even good writing can be condensed without loss of information. I refrain from drawing the obvious conclusion about bad writing.

It seems clear that condensation as well as filtration, performed mechanically or otherwise, should be an essential part of an MIS, and that such a system should be capable of handling much, if not all, of the unsolicited as well as solicited information that a manager receives.

The Manager Needs the Information That He Wants

Most MIS designers "determine" what information is needed by asking managers what information they would like to have. This is based on the assumption that managers know what information they need and want it.

For a manager to know what information he needs he must be aware of each type of decision he should make (as well as does) and he must have an adequate model of each. These conditions are seldom satisfied. Most managers have some conception of at least some of the types of decisions they must make. Their conceptions, however, are likely to be deficient in a very critical way, a way that follows from an important principle of scientific economy: the less we understand a phenomenon, the more variables we require to explain it. Hence, the manager who does not understand the phenomenon he controls plays it "safe" and, with respect to information, wants "everything." The MIS designer, who has even less understanding of the relevant phenomenon than the manager, tries to provide even more than everything. He thereby increases what is already an overload of irrelevant information.

For example, market researchers in a major oil company once asked their marketing managers what variables they thought were relevant in estimating the sales volume of future service stations. Almost seventy variables were identified. The market researchers then added about half again this many variables and performed a large multiple linear regression analysis of sales of existing stations against these variables and found about thirty-five to be statistically significant. A forecasting equation was based on this analysis. An OR team subsequently constructed a model based on only one of these variables, traffic flow, which predicted sales better than the thirty-five variable regression equation. The team went on to explain sales at service stations in terms of the

customers' perception of the amount of time lost by stopping for service. The relevance of all but a few of the variables used by the market researchers could be explained by their effect on such perception.

The moral is simple: one cannot specify what information is required for decision making until an explanatory model of the decision process and the system involved has been constructed and tested. Information systems are subsystems of control systems. They cannot be designed adequately without taking control in account. Furthermore, whatever else regression analyses can yield, they cannot yield understanding and explanation of phenomena. They describe and, at best, predict.

Give a Manager the Information He Needs and His Decision Making Will Improve

It is frequently assumed that if a manager is provided with the information he needs, he will then have no problem in using it effectively. The history of OR stands to the contrary. For example, give most managers an initial tableau of a typical "real" mathematical programming, sequencing, or network problem and see how close they come to an *optimal* solution. If their experience and judgment have any value they may not do badly, but they will seldom do very well. In most management problems there are too many possibilities to expect experience, judgment, or intuition to provide good guesses, even with perfect information.

Furthermore, when several probabilities are involved in a problem the unaided mind of even a manager has difficulty in aggregating them in a valid way. We all know many simple problems in probability in which untutored intuition usually does very badly (e.g., What are the correct odds that 2 of 25 people selected at random will have their birthdays on the same day of the year?). For example, very few of the results obtained by queuing theory, when arrivals and service are probabilistic, are obvious to managers; nor are the results of risk analysis where the managers' own subjective estimates of probabilities are used.

The moral: It is necessary to determine how well managers can use needed information. When, because of the complexity of the decision process, they can't use it well, they should be provided with either decision rules or performance feed-back so that they can identify and learn from their mistakes. More on this point later.

More Communication Means Better Performance

One characteristic of most MIS's which I have seen is that they provide managers with better current information about what other managers and their departments and divisions are doing. Underlying this provision is the belief that better interdepartmental communication enables managers to coordinate their decisions more effectively and hence improves the organization's overall performance. Not only is this not necessarily so, but it seldom is so. One would hardly expect two competing companies to become more cooperative because

the information each acquires about the other is improved. This analogy is not as far fetched as one might first suppose. For example, consider the following very much simplified version of a situation I once ran into. The simplification of the case does not affect any of its essential characteristics.

A department store has two "line" operations: buying and selling. Each function is performed by a separate department. The Purchasing Department primarily controls one variable: how much of each item is bought. The Merchandising Department controls the price at which it is sold. Typically, the measure of performance applied to the Purchasing Department was the turnover rate of inventory. The measure applied to the Merchandising Department was gross sales: this department sought to maximize the number of items sold times their price.

Now by examining a single item let us consider what happens in this system. The merchandising manager, using his knowledge of competition and consumption, set a price which he judged would maximize gross sales. In doing so he utilized price-demand curves for each type of item. For each price the curves show the expected sales and values on an upper and lower confidence band as well. (See Figure 1.) When instructing the Purchasing Department how many items to make available, the merchandising manager quite naturally used the value on the upper confidence curve. This minimized the chances of his running short which, if it occurred, would hurt his performance. It also maximized the chances of being over-stocked but this was not his concern, only the purchasing manager's. Say, therefore, that the merchandising manager initially selected price P_1 and requested that amount Q_1 be made available by the Purchasing Department.

In this company the purchasing manager also had access to the price-demand curves. He knew the merchandising manager always ordered optimistically.

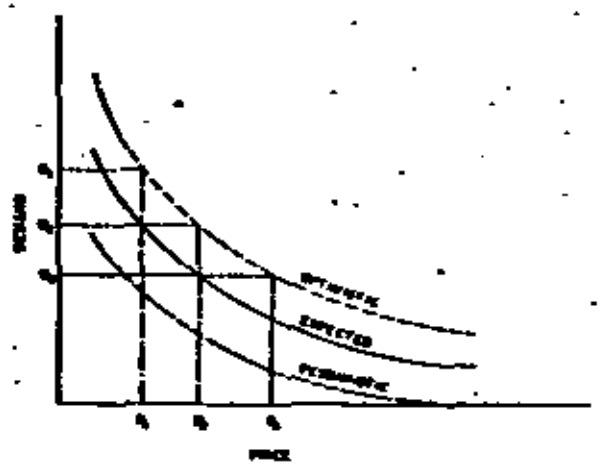


FIGURE 1. Price-demand curve

Therefore, using the same curve he read over from Q_1 to the upper limit and down to the expected value from which he obtained Q_2, the quantity he actually intended to make available. He did not intend to pay for the merchandising manager's optimism. If merchandising ran out of stock, it was not his worry. Now the merchandising manager was informed about what the purchasing manager had done so he adjusted his price to P_2. The purchasing manager in turn was told that the merchandising manager had made this readjustment so he planned to make only Q_2 available. If this process—made possible only by perfect communication between departments—had been allowed to continue, nothing would have been bought and nothing would have been sold. This outcome was avoided by prohibiting communication between the two departments and forcing each to guess what the other was doing.

I have obviously caricatured the situation in order to make the point clear: when organizational units have inappropriate measures of performance which put them in conflict with each other, as is often the case, communication between them may hurt organizational performance, not help it. Organizational structure and performance measurement must be taken into account before opening the flood gates and permitting the free flow of information between parts of the organization. (A more rigorous discussion of organizational structure and the relationship of communication to it can be found in [1].)

A Manager Does Not Have to Understand How an Information System Works, Only How to Use It

Most MIS designers seek to make their systems as innocuous and unobtrusive as possible to managers lest they become frightened. The designers try to provide managers with very easy access to the system and assure them that they need to know nothing more about it. The designers usually succeed in keeping managers ignorant in this regard. This leaves managers unable to evaluate the MIS as a whole. It often makes them afraid to even try to do so lest they display their ignorance publicly. In failing to evaluate their MIS, managers delegate much of the control of the organization to the system's designers and operators who may have many virtues, but managerial competence is seldom among them.

Let me cite a case in point. A Chairman of a Board of a medium-size company asked for help on the following problem. One of his larger (decentralized) divisions had installed a computerized production-inventory control and manufacturing-manager information system about a year earlier. It had acquired about $2,000,000 worth of equipment to do so. The Board Chairman had just received a request from the Division for permission to replace the original equipment with newly announced equipment which would cost several times the original amount. An extensive "justification" for so doing was provided with the request. The Chairman wanted to know whether the request was really justified. He admitted to complete incompetence in this connection.

A meeting was arranged at the Division at which I was subjected to an extended and detailed briefing. The system was large but relatively simple. At the heart of it was a reorder point for each item and a maximum allowable

stock level. Reorder quantities took lead-time as well as the allowable maximum into account. The computer kept track of stock, ordered items when required and generated numerous reports on both the state of the system it controlled and its own "actions."

When the briefing was over I was asked if I had any questions. I did. First I asked if, when the system had been installed, there had been many parts whose stock level exceeded the maximum amount possible under the new system. I was told there were many. I asked for a list of about thirty and for some graph paper. Both were provided. With the help of the system designer and volumes of old daily reports I began to plot the stock level of the first listed item over time. When this item reached the maximum "allowable" stock level it had been reordered. The system designer was surprised and said that by sheer "luck" I had found one of the few errors made by the system. Continued plotting showed that because of repeated premature reordering the item had never gone much below the maximum stock level. Clearly the program was confusing the maximum allowable stock level and the reorder point. This turned out to be the case in more than half of the items on the list.

Next I asked if they had many paired parts, ones that were only used with each other; for example, matched nuts and bolts. They had many. A list was produced and we began checking the previous day's withdrawals. For more than half of the pairs the differences in the numbers recorded as withdrawn were very large. No explanation was provided.

Before the day was out it was possible to show by some quick and dirty calculations that the new computerized system was costing the company almost $150,000 per month more than the hand system which it had replaced, most of this in excess inventories.

The recommendation was that the system be redesigned as quickly as possible and that the new equipment not be authorized for the time being.

The questions asked of the system had been obvious and simple ones. Managers should have been able to ask them but—and this is the point—they felt themselves incompetent to do so. They would not have allowed a handoperated system to get so far out of their control.

No MIS should ever be installed unless the managers for whom it is intended are trained to evaluate and hence control it rather than be controlled by it.

A Suggested Procedure for Designing an MIS

The erroneous assumptions I have tried to reveal in the preceding discussion can, I believe, be avoided by an appropriate design procedure. One is briefly outlined here.

1. Analysis Of The Decision System

Each (or at least each important) type of managerial decision required by the organization under study should be identified and the relationships between them should be determined and flow-charted. Note that this is *not* necessarily the same thing as determining what decisions are made. For example, in one company I found that make-or-buy decisions concerning parts were made only at the time when a part was introduced into stock and was never subsequently reviewed. For some items this decision had gone unreviewed for as many as twenty years. Obviously, such decisions should be made more often; in some cases, every time an order is placed in order to take account of current shop loading, underused shifts, delivery times from suppliers, and so on.

Decision-flow analyses are usually self-justifying. They often reveal important decisions that are being made by default (e.g., the make-buy decision referred to above), and they disclose interdependent decisions that are being made independently. Decision-flow charts frequently suggest changes in managerial responsibility, organizational structure, and measures of performance which can correct the types of deficiencies cited.

Decision analyses can be conducted with varying degrees of detail, that is, they may be anywhere from coarse to fine grained. How much detail one should become involved with depends on the amount of time and resources that are available for the analysis. Although practical considerations frequently restrict initial analyses to a particular organizational function, it is preferable to perform a coarse analysis of all of an organization's managerial functions rather than a fine analysis of one or a subset of functions. It is easier to introduce finer information into an integrated information system than it is to combine fine subsystems into one integrated system.

2. An Analysis Of Information Requirements

Managerial decisions can be classified into three types:

(a) Decisions for which adequate models are available or can be constructed and from which optimal (or near optimal) solutions can be derived. In such cases the decision process itself should be incorporated into the information system thereby converting it (at least partially) to a control system. A decision model identifies what information is required and hence what information is relevant.

(b) Decisions for which adequate models can be constructed but from which optimal solutions cannot be extracted. Here some kind of heuristic or search procedure should be provided even if it consists of no more than computerized trial and error. A simulation of the model will, as a minimum, permit comparison of proposed alternative solutions. Here too the model specifies what information is required.

(c) Decisions for which adequate models cannot be constructed. Research is required here to determine what information is relevant. If decision making cannot be delayed for the completion of such research or the decision's effect is not large enough to justify the cost of research, then judgment must be used to "guess" what information is relevant. It may be possible to make explicit the implicit model used by the decision maker and treat it as a model of type (b).

In each of these three types of situation it is necessary to provide feedback by comparing actual decision outcomes with those predicted by the model or decision maker. Each decision that is made, along with its predicted outcome,

should be an essential input to a management control system. I shall return to this point below.

3. Aggregation Of Decisions

Decisions with the same or largely overlapping informational requirements should be grouped together as a single manager's task. This will reduce the information a manager requires to do his job and is likely to increase his understanding of it. This may require a reorganization of the system. Even if such a reorganization cannot be implemented completely what can be done is likely to improve performance significantly and reduce the information loaded on management.

4. Design Of Information Processing

Now the procedure for collecting, storing, retrieving, and treating information can be designed. Since there is a voluminous literature on this subject I shall leave it at this except for one point. Such a system must not only be able to answer questions addressed to it; it should also be able to answer questions that have not been asked by reporting any deviations from expectations. An extensive exception-reporting system is required.

5. Design Of Control Of The Control System

It must be assumed that the system that is being designed will be deficient in many and significant ways. Therefore it is necessary to identify the ways in which it may be deficient, to design procedures for detecting its deficiencies, and for correcting the system so as to remove or reduce them. Hence the system should be designed to be flexible and adaptive. This is little more than a platitude, but it has a not-so-obvious implication. No completely computerized system can be as flexible and adaptive as can a man-machine system. This is illustrated by a concluding example of a system that is being developed and is partially in operation. (See Figure 2.)

The company involved has its market divided into approximately two hundred marketing areas. A model for each has been constructed as is "in" the computer. On the basis of competitive intelligence supplied to the service marketing manager by marketing researchers and information specialists he and his staff make policy decisions for each area each month. Their tentative decisions are fed into the computer which yields a forecast of expected performance. Changes are made until the expectations match what is desired. In this way they arrive at "final" decisions. At the end of the month the computer compares the actual performance of each area with what was predicted. If a deviation exceeds what could be expected by chance, the company's OR Group then seeks the reason for the deviation, performing as much research as is required to find it. If the cause is found to be permanent the computerized model is adjusted appropriately. The result is an adaptive man-machine system whose precision and generality is continuously increasing with use.

Finally it should be noted that in carrying out the design steps enumerated above, three groups should collaborate: information systems specialists, operations researchers, *and managers*. The participation of managers in the design of a system that is to serve them, assures their ability to evaluate its performance by comparing its output with what was predicted. Managers who are not willing to invest some of their time in this process are not likely to use a management control system well, and their system, in turn, is likely to abuse them.

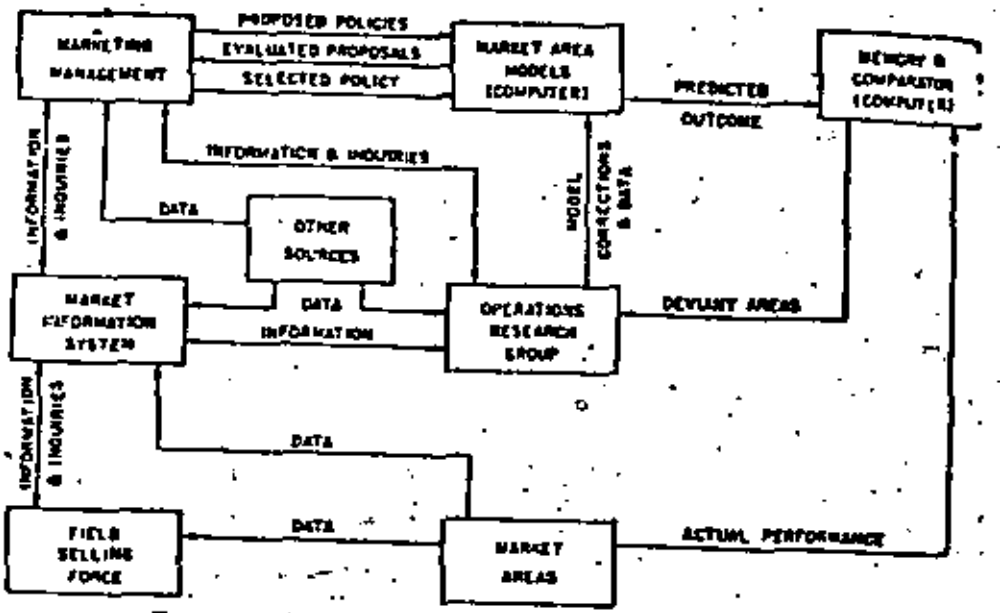


FIGURE 2. Simplified diagram of a market-area control system

References

1. SENGUPTA, S. S., AND ACKOFF, R. L., "Systems Theory from an Operations Research Point of View," *IEEE Transactions on Systems Science and Cybernetics*, Vol. 1 (Sept. 1965), pp. 9-13.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO DATA ARCHITECTURE AND DATA MODEL CONSIDERATIONS

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

Data architecture and data model considerations*

by EDGAR H. SIBLEY and LARRY KERSCHBERG
University of Maryland
College Park, Maryland

ABSTRACT

The Data Base Management System is now a well established part of information systems technology, but the many architectures and their plethora of data models are confusing to both the practitioner and researcher. In the past, attempts have been made to compare and contrast some of these systems, but the greatest difficulty arises in seeking a common basis. This paper attempts to show how a generalized data system (GDS), represented by two different models, could form such a basis; it then proposes that data policy definitions can restrict the GDS to a specialized model, such as a relational or DBTG-like model. Finally, it proposes that this concept forms a better basis for data structure design of specific system applications.

INTRODUCTION

The seventies has seen the acceptance of the database management system (DBMS). Commercial systems and research efforts have proliferated, and the subject has become a major conference topic. However, the potential user is still left with most of the questions that first appeared: Which is the best system? Am I locking myself into one technique or implementation method?

There have been attempts at explaining similarities and differences in the basic classes of systems,[1,2] debate on the effectiveness of different data models,[3] and description of the selection and acquisition process,[4] but confusion remains.

Possibly the reason for difficulty is:

1. The topic is complex. DBMS exist, but they are so different that they defy simple comparison. They also run the gamut of size and sophistication.
2. They differ in methodology of data modelling, retaining, and querying, as well as their internal storage. Testing is expensive: some representative system must be implemented on several DBMS for comparison, or difficult simulations[5] are needed.

Further problems arise in large scale database research and there is need for a common basis[6] as a formalism for describing such systems. Methods of defining the functionality of DBMS include set theory and graph theory constructs.[7] This paper attempts to define such a common basis, and show how it can be used to compare models.

DEVELOPMENT OF A FRAMEWORK

There are at least three distinct levels in an information system: the information and its structure, the data model, and the storage structuring. Obviously, no short paper can cover all three, and we will concentrate on the data model. However, consideration must be given to the information system/data model interface to set the stage for ways to define a good data model with its need to reflect the way data are interrelated, manipulated, and protected.

A *data model* is a system in which a schema may be defined; the DDLC's definition language[8] is principally a mechanism for defining the names and attributes of data elements, groupings, and relationships, while the definition of policy (integrity, security, efficiency, etc.) is almost an afterthought.

The information system—Data model interface

There is an important interface between the organization view of information and the data model constructed to represent it. This interface is being investigated by researchers who are attempting to define a process for producing a good data base design given a set of user needs or inspirations. Reference 9 is a survey of current techniques.

Complete knowledge of the information system and its data usage characterizes the company. Operating policy, however, summarizes the internal constraints of the organization, and the way that the functional subsystems interact; one author[10] refers to these as operative and directive

* The authors wish to express their gratitude to the U.S. Army Computer Systems Command who, through grant number DAAG29-76-G-0300 (Title: Problems in the Translation and Standardization of Relational and Network Type Data Bases), provided partial support and technical advice.

information structures. Several researchers[14,15] have recently advocated the collection of transactions as a basis for designing logical data structures.

Our goal, however, is to develop a framework in which to study data models, and to incorporate important parameters of the interface into the data model: Data Utilization and Operating Policy. By incorporating these parameters into the data model framework we expect to examine some classes and:

- Explain subtle differences,
- Explore declarative versus procedural aspects, and
- Characterize their "semantics."

Data architecture—A level concept

A recent paper[13] proposes that there are four abstraction levels for data machines and models. These, in increasing abstraction, are:

1. *The Defined and Populated Database:* a fully operational data system, with database defined via some definition language.
2. *The Database System:* it involves no data, but represents a specific system, with its description.
3. *The Data Model:* the data system prior to its application. The class(es) of data structures that may be supported by the system have been fixed, but not used.
4. *Data Model Theory:* a conceptual or generalized data base management system generator, assumed to be able to support all classes of data models.

These levels form a progression: From one to four is abstraction—from four to one is utilization; each level naturally subsumes or subsets the previous one.

As an example, Level 3 may be a Relational Model;[16] i.e., it can support a relational data system, but no other. Then Level 2 might be the implementation of a payroll data definition; i.e., a definition of those items (and their attributes) with procedures making up a relational payroll system. At Level 1, we see sets of payroll tuples; i.e., entries for specific people.

We shall use this concept as a basis for the paper. However, there are special operations performed in going from one level to another, defined as follows:

- Level 4 to 3: Data Policy Definition.
  In this step, the management and information system designers state the major constraints on the operational system. The resulting data model (or database machine) at Level 3 is restricted: only some classes of data structures are now allowed; some types of operation are restricted, allowing privacy or security (policy) decisions to be stated; some actions are performed automatically, allowing validation and integrity (policy) to be stated.
- Level 3 to 2: Data Operation Definition.
  Here, the administration is working with a restricted system in which data structure, some efficiency, and specific policy considerations may be stated: e.g., Data Policy Definition specified a relational system with validation at input, now Data Operation Definition defines a database of 2-tuples involving social security number and name, where the former is a nine digit element: This also involves definition of the procedures for building, maintaining, manipulating and retrieving data.
- Level 2 to 1: Data Population and Utilization.
  Finally the data must be loaded and used.

The data model generator—generalized data system

Level 4 data architecture or system can be viewed as either a theory or a machine: i.e., as either an abstract description of a method, or as a machine implementation of that method. If the concepts of Level 4 can be expressed in set theory, then the "machine" could be either a theoretical or working set processor. In this paper, we discuss two candidates for Level 4 machines, and then show how each may be restricted to Level 3 machines. It is, however, of tantamount importance that these machines be truly general, and this exercise is an attempt to show the need and generality, as well as to illustrate the parts and use of such machines. Obviously, if the Level 4 machine is sufficiently general, it will *cover* all possible data machines at Level 3 (at least relational, hierarchic, and network models). It may be considered a meta-data model or generalized data system (GDS). Furthermore, the use of such a system provides a framework for data model comparison.

The two models discussed here are:

i. The Functional Model[7] and
ii. The Set Theoretic and Extended Set Processor[18-19] modified to allow data policy and data operation definition.

THE FUNCTIONAL MODEL OF DATA

Here we consider the Functional Model of Data as a GDS. First, Level 4 structure is presented, then data policy constraints are shown to add form and structure to the model. The constraints are semantic, and allow the Functional Model to be viewed in restricted cases as either relational or DBTG (network) data models.

Level 4 structure

Level 4 is a meta-data level where the Functional Model of data is viewed as a directed graph: its nodes represent sets and its arcs represent total functions. Nodes are either entity sets or value sets. *Entity Sets* may have any number of incoming or outgoing arcs; *Value Sets* may have only incoming arcs, because "values" are the ultimate logical

representation of information, so no arcs leave value set nodes. A typical Level 4 Functional Model graph is shown in Figure 1.

The definitional facilities for the Level 4 Functional Model consists of three creation and naming operations for value sets, entity sets, and functional specification of an entity set (i.e., specification of functions whose domain is the entity set).

There are also operations for deleting value sets, entity sets, and functions. The deletion operations have the following side-effects:

- Deletion of an entity set also implies deletion of those functions incident on it (both incoming and outgoing);
- Deletion of a value set also implies deletion of those functions incident on it;
- Deletion of a function does not affect its domain and range sets, but some may become isolated nodes and may no longer be relevant.

Data policy definition

Data policy decisions are of the following types:

- The methodology to be used to obtain the data structures.
- The representation of elements in the nodes (i.e., sets) of the Functional Model graph.
- The logical access mechanisms to be supported at Level 3.

While both information and management policy ramifications may be stated in a declarative fashion (CODASYL DDL,[5]), management policies are often enforced by transaction-driven "triggers."[20]

To refine these ideas we first address the ramifications of data policy. The most important decision is the choice of data model methodology; this will determine the richness and complexity of the allowable data structures. The methodology adopted in the Functional Model is semantic predication analysis,[21] a process developed to analyze the semantic structure of sentences.

A predication represents a whole sentence; e.g., an assertion, a command, or a question; it may be decomposed into zero, one, or two arguments and a predicate. Arguments may themselves be predications. "Downgraded predications" may qualify arguments (the semantic equivalent of adjectival clauses) or may modify predicates (the semantic equivalent of adverbial clauses). The lowest semantic level consists of semantic features which serve as atomic semantic description units. Downgraded predications play the role of semantic features of the arguments or predicates that they qualify or modify.

Here we assume that a specification of data interrelationships is available. The information analyst's role is to obtain the corresponding predication structures and map these to the Functional Model (an abstraction process). Consider the statement:

"Companies supply parts to departments in some volume"

The predication structure is shown in Figure 2, where the main predication structure "companies supply parts" is represented by the predication PN_1 with arguments A_1 (COMPANIES), A_2 (PARTS) and predicate P_1 (SUPPLY). The arrow under SUPPLY denotes the direction of the relationship represented by PN_1; it corresponds to the active voice of P_1. PN_2 and PN_3 are downgraded modifying predications representing the indirect object and adverbial

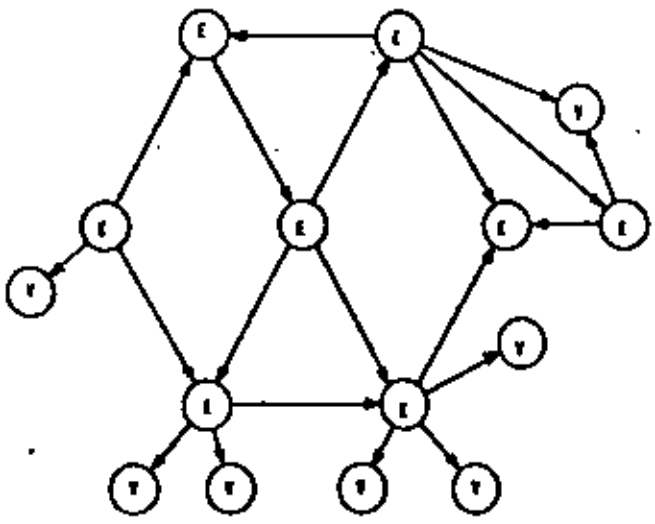


Figure 1—Functional model graph at level 4

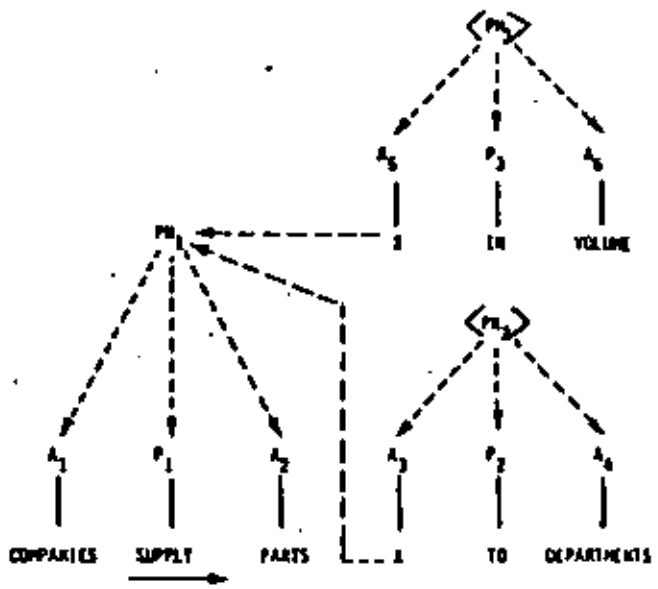


Figure 2—The predication structure for the sentence: "Companies supply parts to departments in some volume"

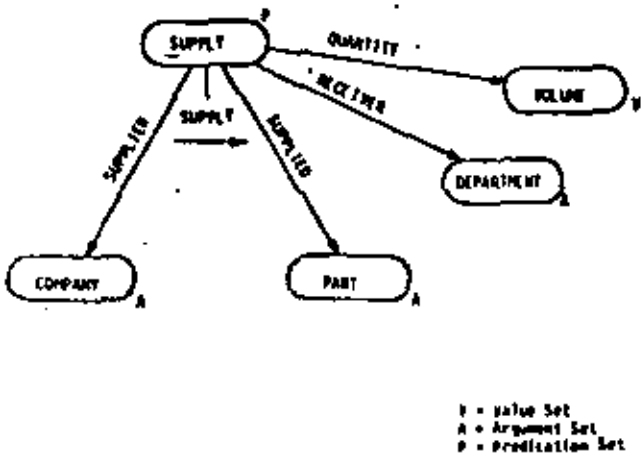


Figure 3—The functional model data structure

information, respectively. The X's refer to P_1, which their corresponding predications modify. Semantic features are not present in this example, but correspond to descriptions of the arguments (COMPANIES, PARTS, DEPARTMENTS and VOLUME). For example, DEPARTMENTS might be characterized by NAME, ADDRESS, and NUMBER.

The choice of the abstraction used to map predication structures to Functional Model data structures is part of data policy. As an example, the model might be restricted as follows:

- Semantic features map to functions whose range sets are value sets.
- Arguments corresponding to "real-world" entities map to named argument sets.
- Predications map to named predication sets, and the arcs pointing to arguments become named functions. Also the predicate and its arrow are attached to the predication set.
- Downgraded predications are represented by functions whose domain is the main predication set and range is either an argument set or a value set.

Figure 3 depicts the Functional Model data structure based on the predication structure of Figure 2 and the above abstraction rules.

Thus the choice of the methodology used to model the organization is one aspect of data policy which induces structure in passing from Level 4 to Level 3. At Level 3 the Functional Model has entity sets classified as either predication sets or argument sets (denoted P and A, respectively) and value sets remain unchanged. Functions perform two roles: a function may provide information about a predication structure or it may represent a semantic feature. The data structures supported by the predication analysis and abstraction process are depicted in Figure 4. All these data structures are possible for the Functional Model, but most applications use only [illegible]

- Functions emanating from a predication set cannot have predication sets as ranges.

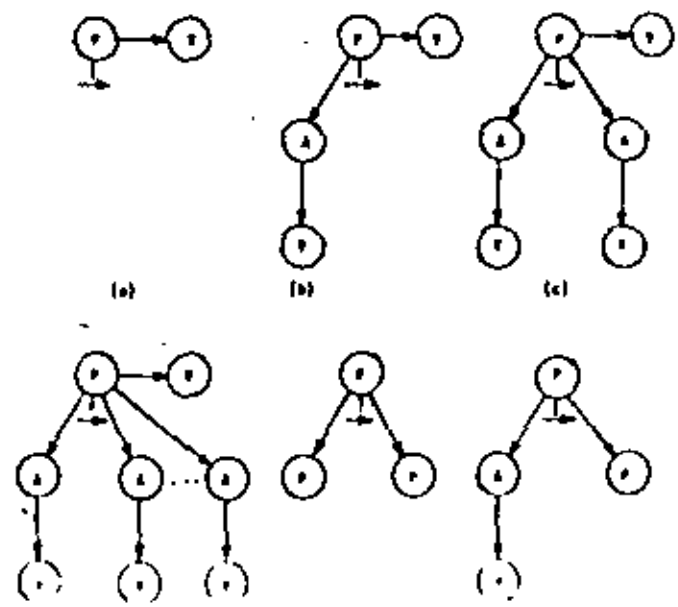
then only cases c and d are admissible. This is precisely the case in the Entity-Relationship model,[9] where predication and argument sets are called relationship sets and entity sets, respectively. In the Functional Model arcs all represent total functions, whereas in the Entity-Relationship model arcs are either 1:N mappings (for entity sets to relationship sets) or functions (for entity sets to value sets).

So far, we have only considered the methodology for obtaining data structures. Although the set types play an important role, the function types are important in expressing logical access mechanisms. Consider a binary relation α from set A to set B, and its representation, R_α, as the set of ordered pairs

$$R_\alpha = \{(a, b) \mid a \in A \;\&\; b \in B \;\&\; a \alpha b\}$$

Obviously, there exists an "inclusion" function, i, which assigns an element (a, b) of R_α to its corresponding element (a, b) of AxB, the Cartesian product of A and B. In addition, there exist functions $f: R_\alpha \rightarrow A$ and $g: R_\alpha \rightarrow B$ such that the diagram of Figure 5 "commutes."[10] In terms of the Functional Model predication structures, the binary relation α corresponds to a predication structure consisting of a predication node (labelled "α"), argument nodes (A and B) and arcs (f and g) (see Figure 5). The actual representation of the elements in the predication set is by means of the set R_α.

The functionality types of f and g are important in modelling the semantics of α: i.e., whether α is a relation, a partial function, or a total function. There are sixteen possible configurations for the predication structure, because f and g may be one-to-one, onto, both of these, or none of these. The most important configurations are



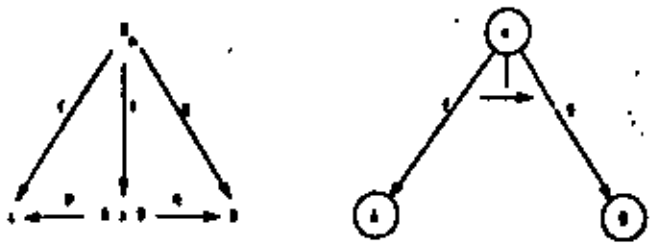


Figure 5—Representation of a binary relation and predication structure

summarized in the following:
Fact:
Let α be a binary relation from A to B with functional specification f and g.
Then:

1. If neither f nor g are one-to-one (1:1), then α is a relation;
2. If f is 1:1, then α is a partial function;
3. If f is 1:1 and onto, then α is a total function;
4. If f is 1:1 and onto, while g is 1:1, then α is a 1:1 function;
5. If f is 1:1 and onto, while g is onto, then α is an onto function;
6. If f and g are both 1:1 and onto, then α is also 1:1 and onto.

The functionality type of f determines whether α is a relation, a partial function, or a total function. If α is a function, then the functionality of g determines whether α is one-to-one, onto, a one-to-one correspondence, or none of these.

In terms of predication structures, a predication set may thus represent a relation or a function, which implies that, in the latter case, α may be represented by an arc. This is indeed true, but it must be considered a Data Policy decision.

One-to-one functions play an important role in accessing a particular element of a set. If the function f in Figure 5 is one-to-one, a particular $a \in A$ will participate in (at most) one ordered pair $(a,b) \in R_\alpha$, so that the second element of the ordered pair may be obtained by evaluating the function g. Thus, any one-to-one function outgoing from a set is a candidate (key) for accessing the elements of the set.

The notion of logical access can be extended to predication sets involving more than two argument sets and various value sets (as in Figures 3 and 4). In this case, the predication node represents an n-ary relationship, where each element may be viewed as an n-tuple of entities and values. For n-ary predication sets, a composite key is precisely the concept in Reference 22.

Finally we illustrate a management policy constraint whose predication structure has another predication node as an argument (e.g., Figure 4(f)). The operational constraint "A company must supply at least three parts to some department during a quarter to remain a valid supplier," could be modelled by a predication structure consisting of a "must" predication set with argument sets COMPANY and SUPPLY (a Figure 3 predication set), and semantic features (functions to value sets) TOTAL-PARTS and QUARTER. Elements of the "must" set might be updated by program on the occurrence of a SUPPLY transaction, but the validation would probably take place at the end of each operating quarter.

The relational model of data

The data policy constraints on the Functional Model predication structures which transform it into the Relational Model are:

- Value sets are *domains*;
- Argument sets and predication sets are *relations* whose elements are represented by *tuples* of values from domains;
- Functions whose range sets are value sets become the *attribute names* of their respective relations;
- Functions from predication sets to argument sets are replaced by the attribute name corresponding to the key (perhaps the composite key) of the argument set.

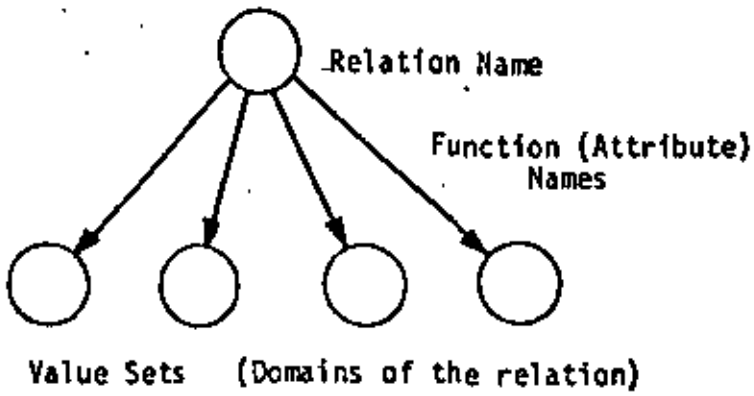
The consequence of these restrictions is to allow only nodes which represent Level 4 entity sets as relations: represented by tuples. The arcs can now only point away from nodes, and their functions are the names of the value sets of each element of the tuple; see Figure 6.

The DBTG data model

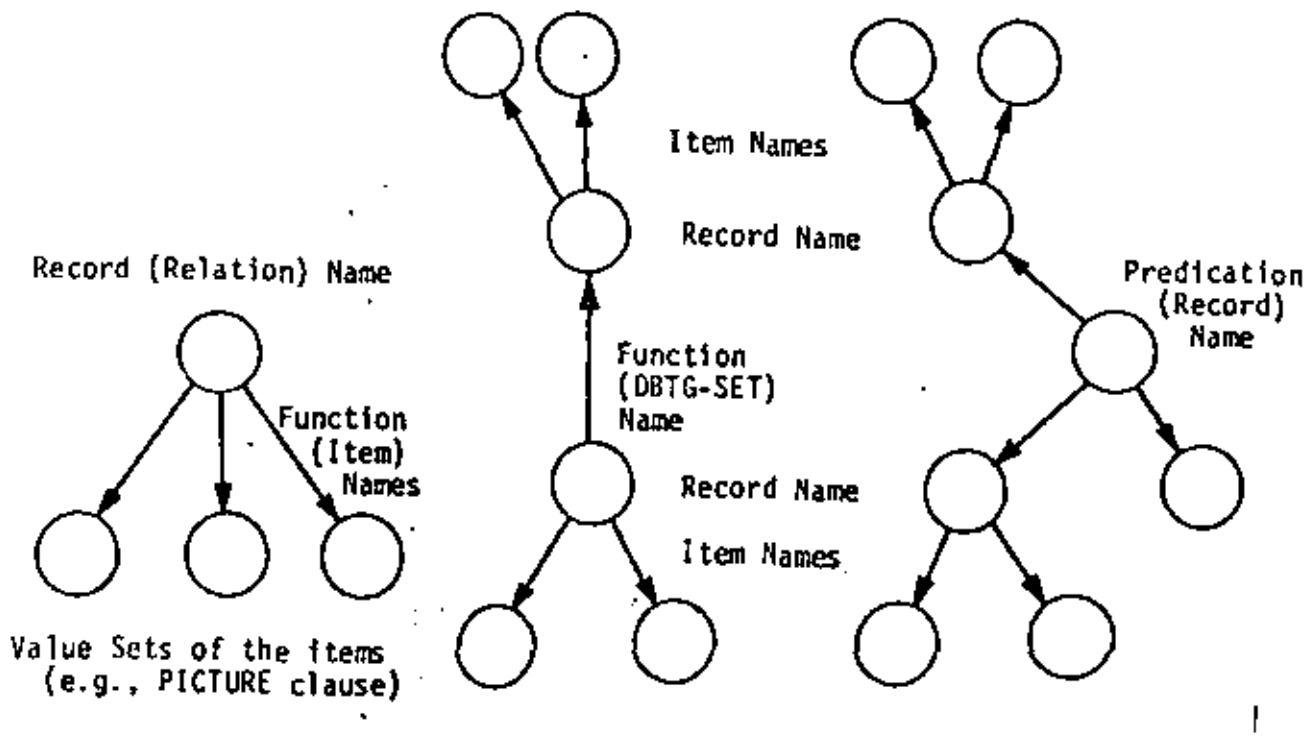
The Functional Model can also be transformed into DBTG type Data Structures by data policy definitions. First, the record will be simplified by ignoring repeating groups—in this case a record is a tuple, and may be represented in the same way as the relational data structure. The DBTG-set, in its simplest form, is a representation of a functional predication (i.e., a predication which is a function). The arrow of the function name is in the opposite direction to the arrow in the equivalent "Bachman" Diagram.[24] For a more complicated DBTG-set structure, there is a predication (a record) between two other relations (also records). This represents an intermediate or link record between two others; these structures have been discussed as linking records between two relations in Reference 25 and termed "associations" in Reference 26.

Whether the model transforms a function name into a DBTG-set or follows the Predication (Record) name model is a data policy decision: the same results may be obtained provided that one record is functionally dependent on the other—but not if the relation is N to M.

The insertion property of a DBTG-set is either AUTOMATIC or MANUAL. Which set has which property is defined at Level 3, but the ability to define these properties of a function is a Data Policy decision, which delimits the Level 4 to 3 structure and defines the DBTG Model. Similarly, the deletion properties (MANDATORY and OP-



a) Relational Data Structures



b) DBTG-like Data Structures

Figure 6—Level 3 data structures in the functional model

TIONAL) are Level 4 to 3 data policy decisions which define the operation of a DBTG model. These two properties are therefore *declaratives* associated with the arcs, similar to the functionality types in a functional model, but obviously having stronger effect. The enforcement of this policy is, of course, procedural.

Comparisons

It is useful to compare and contrast relational and DBTG data policy operations. Relational systems like System R[8] and INGRES[9] use the theory of normal forms to provide good data structures and then utilize these simple structures

through data-name linkages and data operations like JOIN and PROJECT; they apply other data policy such as consistency, integrity, and validation through system modification of the user statement (INGRES), which is immediately initiated, or through system triggers (SYSTEM R) which are transaction driven, but may be delayed. These two implementations differ in the fact that INGRES specifies policy in declaratives with procedural enforcement, while SYSTEM R is procedural both in declaration and enforcement.

The DBTG data policies are both declarative and procedural: some, such as AUTOMATIC, are declarative and apply a primitive function on operation implying a "semantics" at storage of a record; others, the Data Base Procedures, are procedural and are invoked by some trigger: e.g., on update (validation), privacy and security checking.

THE SET THEORETIC DATA MODEL

The set theoretic processor owes its existence to the concept of an extended set. The *extended set* consists of elements which themselves may be conventional sets, sequences, ordered sets, atoms, or even extended sets, but each element is identified by a position-identifier (numeric or mnemonic). The extended set is enclosed in square brackets [ ] while the ordered sets or sequences are in angle brackets ( ). Thus if X is an extended set consisting of elements Y and Z in positions 1 and NEXT, we have:

$$X=[(1,Y),(NEXT,Z)]$$

If we use braces { } to enclose sets, then if Y is the set of the first three integers and Z is the four-tuple consisting of "0,1,0,2" (in order), then:

$$Y=\{1,2,3\}=[(\#,1),(\#,2),(\#,3)]$$

and

$$Z=(0,1,0,2)=[(1,0),(2,1),(3,0),(4,2)].$$

Reference 16 shows that the extended set is a normal extension of set theory and that predicate calculus operations can be defined on the extended set. All normal set operations (union, intersection, difference, etc.) may also be defined, except that operations are position dependent.

Level 4 structure

The model consists of the following objects or elements: Atoms, which represent a number or character string; Sets, composed of any object; Ordered sets or sequences, composed of any object; Extended sets, composed of any object. Of course, no object may contain itself. The other model objects may all be represented as extended sets, position identifiers (which are atoms), and atoms. Commas are used to delimit objects in a sequence or set.

The basic language of the Level 4 data model consists of: predicate calculus expressions; algebraic expressions; assignment (storage) statements; retrieval expressions, with predicates to limit the response; and macros which simplify operations (e.g., Average, Sum).

In order to illustrate the operations of the extended set processor (XSP) at Level 4, a series of operations is given in Figure 7. The first ten operations are all of the "storage by assignment" type: they store ten extended sets, with synonyms (names) ME, YOU, etc. The VALUE function invokes a "copy" operation, which does not necessarily duplicate the string: the result may be represented internally by pointers. The set AUTHORS is therefore made up of the two extended sets ME and YOU.

The assignment operation for X defines a new extended set: it is a simple set containing the names of the two authors of this paper. This particular operation extracts (from the set AUTHORS) the values which have position indicator NAME. Furthermore, the summation operation (SUM) counts the elements (two) of this newly defined set X. The keyword LIST in the find instruction of Figure 7 is used to denote a set of 0 to "N" replications. The names and symbols which have not yet been described, such as PHONE, NAME, AUTHOR-TYPE and PICTURE, have no semantic meaning at Level 4. Later, some will have semantics in defining policy, but *here they are merely symbols*. At Level 4 the XSP is unrestricted. It will store, maintain, retrieve, or manipulate the whole or any part of any extended set. However, the XSP must have operations which can be triggered by events or conditions; these operations, applied in going from Level 4 to Level 3, are the Data Policy definitions.

Data policy definition

There are two types of XSP system applied constraints: static and dynamic. The *static constraint* is one that must always be satisfied: e.g., the data structure class must be hierarchic, or the access to some parts of the database are password protected. Such constraints imply a system action whenever violation occurs, and these may be implemented as special system actions ("compiled into the system") or as interpreted actions, with well defined error response. Most current systems "compile" these constraints (i.e., they "do not support other models of data" or "allow the following types of data protection . . ."). The *dynamic constraint* is one that is satisfied at specific times or for special operations: e.g., the validation of an element is only to be performed at input, or security is to be checked against type of user and type of operation for every access. These constraints are essentially interpreted.

The relational model of data

An example of the definition of allowable data structure classes for a relational model is given in the restrictions of Figure 8. This definition states that all sets are made up of ordered sets which contain "attribute-value" pairs (e.g., the elements ME and YOU in Figure 7 are ordered sets of three pairs). Moreover, the position identifiers shall be

```
ME = [<NAME,SIBLEY>,<SS#,017|32|7992>,<PHONE,(301)262-7138>]

YOU = [<PHONE,(301)937-7726>,<NAME,KERSCHBERG>,<SS#,294|36|4321>]

IT = [<TITLE,DATA ARCHITECTURE ETC.>,<AUTHOR,(VALUE(ME),VALUE(YOU))>]

AUTHOR-TYPE = [<NAME,NAME-TYPE>,<PHONE,PHONE-TYPE>,<SS#,SS#-TYPE>]

NAME-TYPE = (PICTURE X(25 MAX))

PHONE-TYPE = (PICTURE '('999')'999'-'9999)

SS#-TYPE = (PICTURE 999'/'99'/'9999)

ARTICLE-TYPE = [<1,TITLE-TYPE>],

TITLE-TYPE = (PICTURE X (UNRESTRICTED))

AUTHORS = (VALUE(ME),VALUE(YOU))

PRINT (NAME-TYPE)

X = (NAME | SET = AUTHORS)

Y = SUM(X)

Z = SUM ((AUTHOR|SET=IT))

X1 = (NAME|SET=AUTHORS AND SS# NOT='017|32|7992')

AUTHORSHIP-TYPE = [<AUTHOR,LIST (AUTHOR-TYPE)>,<ARTICLE,ARTICLE-TYPE>]

AUTHORSHIP = (VALUE(IT))
```

Figure 7—Some operations of an XSP at Level 4

POLICY RESTRICTIONS OF XSP.

OBJECTS: ATOM, ORDERED-SET, SET.

MEMBERSHIP: MEMBER (ORDERED-SET) IS <POSITION-ID, ATOM>,
MEMBER (SET) IS <#, ORDERED-SET>.

CONSTRAINT: ALL POSITION-ID (ORDERED-SET) IS MEMBER (POSITION-ID
(ORDERED-SET-DEFINITION)).

LEVEL 3: DEFINITION WITHIN RESTRICTION.

SET OBJECT. AUTHORS.

ORDERED-SET OBJECT. AUTHOR-TYPE.

ATOM OBJECT. NAME-TYPE, PHONE-TYPE, SS#-TYPE.

TIMING.

APPLY PHONE-TYPE, SS#-TYPE ON INPUT.

APPLY NAME-TYPE ON OUTPUT, INPUT.

APPLY AUTHOR-TYPE ALWAYS.

An Invalid DEFINITION would be:

SET OBJECT. AUTHORSHIP.

ORDERED-SET OBJECT. AUTHORSHIP-TYPE.

... (See Figure 4.1: this is not a first
Normal Form definition)..

Figure 8— Policy statements and level 3 definition for an XSP: Example 1: Relational model

found within a definition. Thus, if we consider the Level 3 definitions of Figure 8: AUTHOR-TYPE defines the position identifiers (NAME, PHONE, SS#), then ME and YOU conform to this type of ordered set. Moreover, the set AUTHORS in Figure 7 now conforms to the definition AUTHORS in Figure 8.

It is now obvious that Figure 7 contains definitions of extended sets which can *apply* (or be applied) with semantics at Level 3. We class the elements by statements in Figure 8 which show the definition that AUTHORS contains elements (tuples) which comply with the (Figure 7) AUTHOR-TYPE definition, while the (validation) criteria of the atoms SS#-TYPE, etc., are applied on input, with NAME-TYPE checking also on output. Moreover, because some extended set operations might allow generation of invalid sets during valid operations, the AUTHOR-TYPE criterion is applied on all operations (this may be a duplicative statement, depending on the overall constraint . . . ALL POSITION-ID . . . etc., being universally applied, or "compiled into the system").

If we gave XSP, the definition for AUTHORSHIP-TYPE, an error must occur, because the definition in Figure 7 allows non-atomic elements in the ordered set (due to LIST).

It will be seen that the definitions do not truly reflect the first requirement, and that the second can be considered a triggered or system applied procedure (the result of an operation depending on the functional properties of the DBTG-set, etc.). Thus the restrictions are not correct. By further refining the models, it should be possible to determine true similarities and differences. However, it is first necessary to add the operations and their mappings—a non-trivial task.

```
POLICY RESTRICTIONS OF XSP.

OBJECTS:      ATOM, RECORD, DBTG-SET, SYSTEM-SET.
MEMBERSHIP:   MEMBER (RECORD) IS <POSITION-ID, ATOM>,
              MEMBER (DBTG-SET) IS <1, RECORD OR
              <2, LIST (RECORD)> .
              MEMBER (SYSTEM-SET) IS DBTG-SET.
CONSTRAINT:   ALL POSITION-ID (RECORD) IS MEMBER
              (POSITION-ID (RECORD-DEFINITION)),
              ...
```

The DBTG data model

In dealing with the definition of a DBTG-like structure, one is faced with some prior decisions. Using the definition of Reference 25, a DBTG-set consisting of an owner record (A_i) and three member records (B_u, B_u', B_u''), in that order, will be represented as

$$(A_i, (B_u, B_u', B_u''))$$

The restrictions imposed in an earlier section on DBTG-records is applied here also: no repeating groups are allowed. Thus the definition of a record in Figure 9 follows that of an ordered set in Figure 8. The definitions therefore are special only in their inclusion of membership in DBTG-sets.

Data Policy is represented by the ability to have AUTOMATIC operation of DBTG-set inclusion on storing a predefined record.

```
LEVEL 3:  DEFINITION WITHIN RESTRICTION.
    SYSTEM-SET OBJECT.  AUTHORSHIP
    DBTG-SET OBJECT.  AUTHORSHIP-TYPE.
    RECORD OBJECT.  ARTICLE-TYPE, AUTHOR-TYPE.
    ATOM OBJECT.  TITLE-TYPE, NAME-TYPE, PHONE-TYPE, SS#-TYPE.
    TIMING.
    APPLY AUTOMATIC TO ARTICLE-TYPE ON STORE.
```

Figure 9—Policy statements and Level 3 definition for an XSP: Example 2: DBTG model

Comparisons

It has been suggested that the implementation of a relational structure in a DBTG system is a matter of:

1. Only allowing system owned sets (DBTG termed this a "SINGULAR SET");
2. Removing concepts of database procedures, inclusion and deletion properties (AUTOMATIC, MANDATORY, etc);
3. Making keys unique (using CALC with a DUPLICATES NOT ALLOWED clause);
4. Allowing new macros like JOIN and PROJECTION on (mathematical) sets of records.

CONCLUSIONS

Work with two GDS (Functional and Extended Set Models) leads us to the following conclusions:

1. The model of an enterprise information structure may be defined independently of the GDS used for its implementation.
2. If the GDS is "universal" it may store any information structure (both data syntax and semantics) in terms of its primitives.
3. The specialization of the GDS to a data model like supported by most current research and commercial DBMS involves Data Policy definition. Using this, a GDS may be restricted to perform as one or specific data models.
4. The process of mapping from an information struc within a GDS to a data structure within a traditional

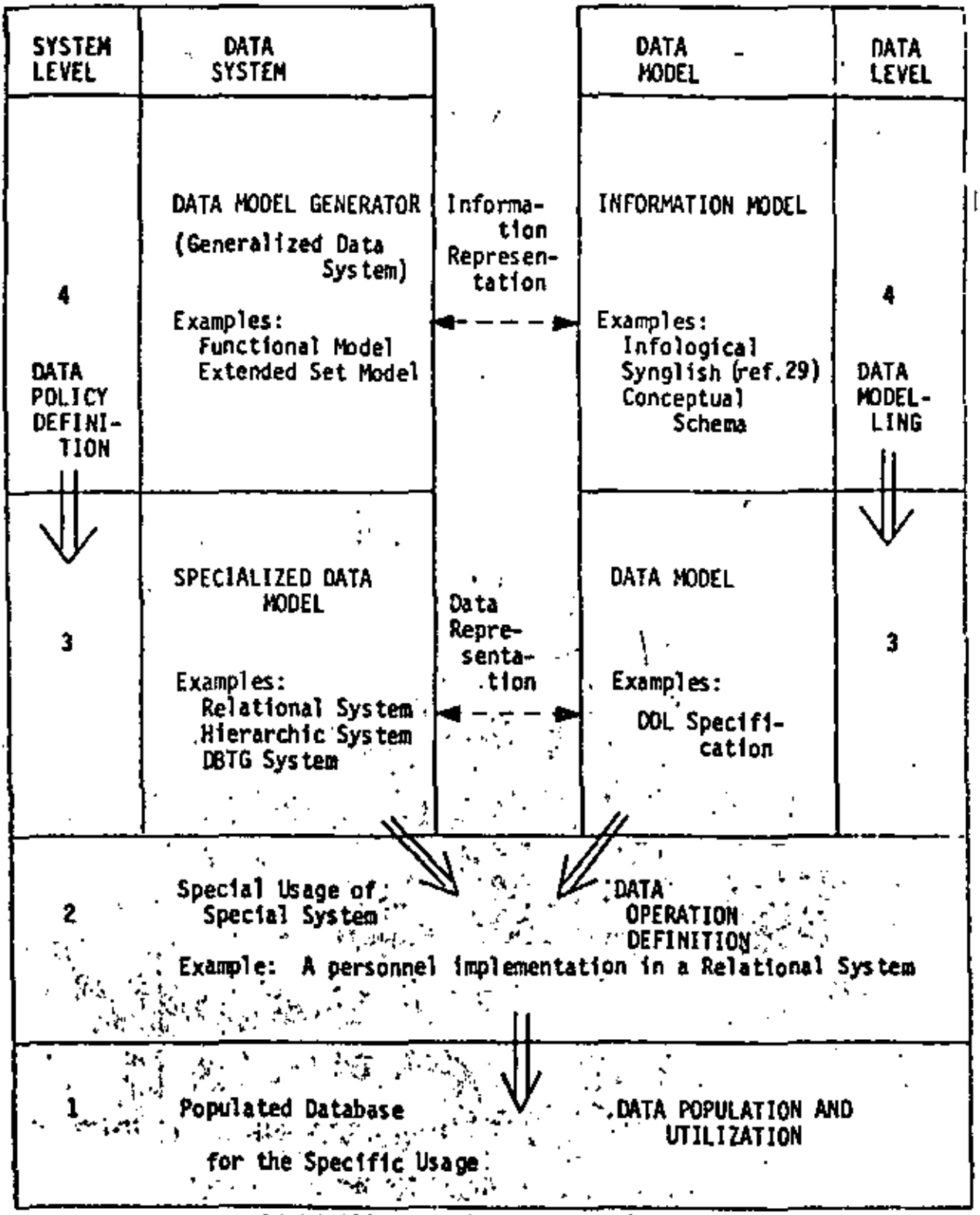


Figure 10—The parallelism of data policy and modelling

DBMS follows the restrictions of the Data Policy definition. Consequently there must be a correspondence between Data Policy restriction and *data modelling* (i.e., passing from information structure to data structure). This process is diagrammed in Figure 10.

Most information analysts already have a specialized data model (e.g., DBTG systems) in mind when constructing their information model. Thus, they take the Data Modelling route in Figure 10. We suggest that the correct general) process is to express conceptual infor

tures in the GDS and to "tune" the information structures to data structures by means of Data Policy decisions. In other words, it is advantageous to represent information at Level 4 so that all semantics of the information are retained.

Data Policy definitions impose added structure (with restrictions) on allowable data structures. Tradeoffs will then make it easier to consider the losses in representation of information structures in the supported data structures.

REFERENCES

1. Sibley, E. H., Guest Editor: "Special Issue on Data Base Management Systems," *ACM Computing Surveys*, Vol. 8, No. 1, March 1976, pp. 1-11
2. Kerschberg, L., A. Klug and D. Tsichritzis, "A Taxonomy of Data Models," *Proceedings Second International Conference on Very Large Data Bases*, Brussels, September 1976.
3. Rustin, R., Editor: *ACM SIGMOD 1974 Workshop on Data Description, Access, and Control*, "Data-Structure-Set versus Relational," May 1974, pp. 144.
4. CODASYL Systems Committee: "The Selection and Acquisition of Data Base Management Systems," Published by ACM, New York and IAG, Amsterdam, March 1976, pp. 232.
5. Reiter, A., "Data Models for Secondary Storage Representations," *Proceedings 1st International Conference on Very Large Data Bases*, ACM, Sept. 1975, pp. 87-119.
6. Hardgrave, W. T., and E. H. Sibley, "Database Research: Some Comments on Future Directions," *SIGMOD FDT 7*, pp. 3-4, 1975.
7. Kerschberg, L. and J. E. S. Pacheco, "A Functional Data Base Model," Computer Science Monograph, Pontificia Universidade Catolica do Rio de Janeiro, February, 1976, also available as Technical Report 13, Dept. of Information Systems Management, University of Maryland, 1976.
8. CODASYL Data Description Language Committee, "Data Description Language Journal of Development," National Bureau of Standards Handbook 113, Washington, 1973, pp. 136.
9. Novak, D. O. and J. P. Fry, "The State of the Art of Database Design," *Proceedings Fifth Texas Conference on Computing Systems*, Austin, 1976, pp. 30-38.
10. Langefors, B., "Theoretical Aspects of Information Systems for Management," *Proceedings IFIP Congress 74*, pp. 937-943.
11. Rond, D. Sheppard, "Data Base Design Methodology Parts I and II," AUERBACH Publishers Inc., 1976.
12. Kahn, B. K., "A Method for Describing the Information Required by the Data Base Design Process," *Proceedings International ACM-SIGMOD Conference on Management of Data*, Washington, D.C., 1976, pp. 53-64.
13. Rothnie, J. B. and W. T. Hardgrave, "Data Model Theory: A Beginning" *Proceedings Fifth Texas Conference on Computing Systems*, Austin, 1976.
14. Codd, E. F., "A Relational Model of Data for Large Shared Data Banks," *Comm. ACM*, 13, June 1970, pp. 377-387.
15. Childs, D. L., "Description of a Set-theoretic Data Structure," *AFIPS Conf. Proc.*, Vol. 33, Part 1, AFIPS Press, Montvale, N.J., 1968, pp. 557-564.
16. Childs, D. L., "Feasibility of a Set-theoretic Data Structure: A General Structure Based on a Reconstructed Definition of Relation," *IFIP Congress 1968*, North Holland, Amsterdam, 1968, pp. 420-430.
17. Childs, D. L., "Extended Set Theory: A Formalism for the Design, Implementation, and Operation of Information Systems," Volume IV, *Current Trends on Programming Methodology*, edited by R. T. Yeh, Prentice-Hall,
18. Hardgrave, W. T., "A Technique For Implementing a Set Processor," *Proc. ACM Conference on Data: Abstraction, Definition, and Structure*, SIG-PLAN Notices, March 1976.
19. Hardgrave, W. T., "Set Processing: A Tool for Data Management," *Proc. ACM/NBS Fifteenth Annual Technical Symposium*, June 1976.
20. Eswaran, K., "Aspects of a Trigger Subsystem in an Integrated Database System," *Proceedings Second International Conference on Software Engineering*, San Francisco, 1976, pp. 243-250.
21. Leech, G., *Semantics*, Penguin Books Ltd., Middlesex, England, 1974.
22. Chen, P., "The Entity-Relationship Model—Toward a Unified View of Data," *ACM Transactions on Database Systems*, Vol. 1, No. 1, March 1976, pp. 9-36.
23. MacLane, S. and G. Birkhoff, *Algebra*, Macmillan Co., 1968.
24. Bachman, C. W., "Data Structure Diagrams," *Data Base*, ACM SIGBDP Newsletter No. 1, 2, Summer 1968.
25. Sibley, E. H., "On the Equivalences of Data Based Systems," *ACM SIGMOD 1974 Workshop on Data Description, Access, and Control*, May 1974, pp. 43-78.
26. Schmid, H. A. and J. R. Swenson, "On the Semantics of the Relational Data Model," *Proceedings International ACM-SIGMOD Conference on Management of Data*, 1975.
27. Astrahan, M. M. et al., "System R: A Relational Approach to Data Base Management," *ACM Transactions on Database Systems*, Vol. 1, No. 2, 1976.
28. Stonebraker, M., "High Level Integrity Assurance in Relational Data Base Management Systems," *Proceedings of the 1975 ACM-SIGMOD Workshop*.
29. Kerschberg, L., E. A. Ozkarahan, and J. E. S. Pacheco, "A Synthetic English Query Language for a Relational Associative Processor," *Proceedings Second International Conference on Software Engineering*, San Francisco, 1976, pp. 305-319.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

WHY RESTRICT THE MODELING CAPABILITY OF CODASYL DATA STRUCTURE SETS,

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

Why restrict the modelling capability of codasyl data structure sets?

by CHARLES W. BACHMAN
Honeywell Information Systems
Billerica, Massachusetts

ABSTRACT

Several issues have been raised concerning changes to the capabilities of the CODASYL Data Description Language specifications for data structure sets. The paper argues against new restrictions suggested and for removal of existing restrictions. The issues are:

allow recursive set declaration
keep multiple member declaration
allow alternate owner declarations

The concept of "record-roles" is introduced to justify the need for these capabilities. The expanded capabilities described offer an alternate means of achieving the same end result without the need to introduce the "record-role" into the CODASYL Data Description Language.

INTRODUCTION

The concept of data structure sets has been well established through the publicity and use of I-D-S, the Honeywell Integrated Data Store system.[1-3] In recent years this concept has been adopted by the various CODASYL committees and imbedded in the CODASYL Data Description Language[4] and the COBOL Data Manipulation Language.[5] A number of hardware and software suppliers have implemented the data structure sets of these languages (DDL/DML) as part of their systems. They include:

IDMS	(Cullinane for IBM 360/370)
I-D-S	(Honeywell GE200, GE400, GE600, H6000)
I-D-S II	(Honeywell for H66, H64)
EDMS	(Xero Sigma 6/7/8)
DMS 1100	(Univac for Univac 1100 Series)
PHOLAS	(Philips for Unidata 7000, P1000)
PHOLAS	(Siemens for S4004)

Other implementations have been reported for CDC and DEC.

The capabilities of the data structure set, as developed in I-D-S and now defined in both the CODASYL DDL and the COBOL DML, provide for set-type declarations which:

(1) restrict the record type declared as owner to be different from any of the record types declared as member.
(2) permit declaration of one or more record types to serve as member records of an occurrence of a set type, and
(3) restrict to one the number of record types which can be declared to serve as owner records of occurrences of a set type.

THE PROPOSALS

This paper is a refinement of a working paper written in response to an assignment accepted at the IFIP-TC2 meeting on Data Description Languages held in Namur, Belgium in January 1973. There were three closely related proposals for changes to the CODASYL DDL discussed at that meeting. These proposals relate directly to three points enumerated in the prior paragraph. Assignments were given to defend a number of such proposals. The proposals interrelated and my working paper treats them as a single concept.

The first proposal, which was unanimously supported at the meeting, was to remove the restriction that the record-type declared as owner could not also be one of those declared as member. I strongly concurred with this proposal as a removal of an unnecessary restriction.

The second proposal was to add a restriction that only one record-type could be declared as member of a particular set-type. This proposal received mixed support. I strongly disagreed with this proposal, for essentially the same reasons that I support the first and third proposals. It adds an unnecessary restriction.

The third proposal was to remove the restriction that only one record-type can serve as owner of a particular set-type. This proposal received scant attention. The meeting did not express an opinion on the subject. I strongly recommended it, as I have done to the DBTG at least seven years ago. It is the removal of a restriction.

ARGUMENTS

There are several arguments for permitting a set-type to permit the record-type declared as member to be the same as the record-type declared as owner. There is a specific argument which will be treated first and then a general argument which relates to all three proposals mentioned earlier.

The specific argument treats the need for tree-like data structures, catalogues, organization structures and parsing trees. For these structures, it is necessary to support recursive sets, which provides the capability to build trees, with branches which have branches, which have branches, etc. An example of this is illustrated in Figure 1.

If all the straight lines in Figure 1 are considered to be "branches," then this structure can be built with a single record type and a single set-type. However, the membership of the "branch" record-type in the "fork" set-type must not be mandatory. A record declared to be as the lowest level branch is never a member of a "fork" set-occurrence. This special branch is characterized as the trunk. Figure 2 is a data structure diagram[4] which illustrates the branch/fork structure.

In this data structure diagram, the "fork" set-type is illustrated with a broken line, meaning that the "branch" record-occurrences are sometimes members of it. That is, they are members if they are not the "trunk" branch of the tree. Note that all branch record-occurrences, whether or not they are the trunk, own "fork" set-occurrence with zero, one, or more subordinate "branches" records.

The third Namur proposal, which supports alternative owners in data structure sets, would permit the trunk branch to be treated as a distinctly different type of entity. Figure 3 is a redrawing of Figure 1.

It illustrates a tree with one trunk and many forks and branches. When this is drawn as a data structure diagram, the illustration of Figure 4 is developed. This structure has some advantages.

The "branch" record-occurrences can now be treated as mandatory members of the "fork" set-type, i.e., no branches are floating in the air. This can be very important

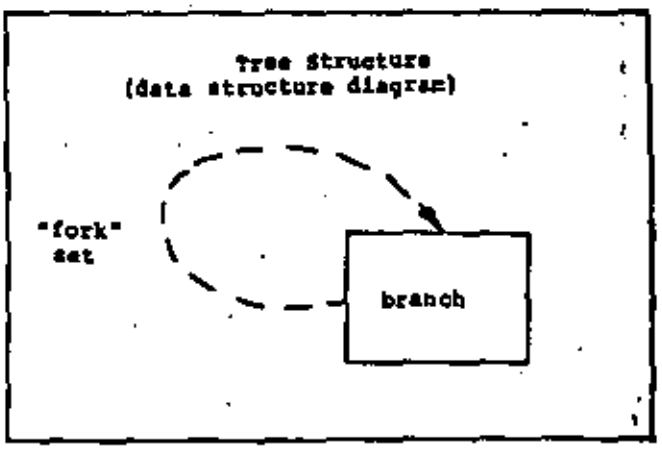


Figure 2

from a naming point of view, as each branch needs a for its "branch name" while the trunk does not need such a name. Branches are frequently named with an articula grammar. All the branch names at a single fork of the tree must have unique local names. The higher level branc (i.e., the branches which are farther from the trunk), uniquely named by concatenating their unique local n to the tree unique name of the branch immediately below. This is expressed below in BNF (Backus/Nauer/Form)

⟨branch-name⟩::=⟨character-string⟩ |
⟨branch-name⟩⟨articulation-character⟩
⟨character-string⟩
⟨character-string⟩::=⟨character⟩ | ⟨character-string⟩
⟨character⟩
⟨character⟩::=a|b|c|.....|z|0|1|.....|9
⟨articulation-character⟩::=any symbol which is not a blank or a character

In our information systems today, there are many examples of tree structures. In catalog (file) systems, we

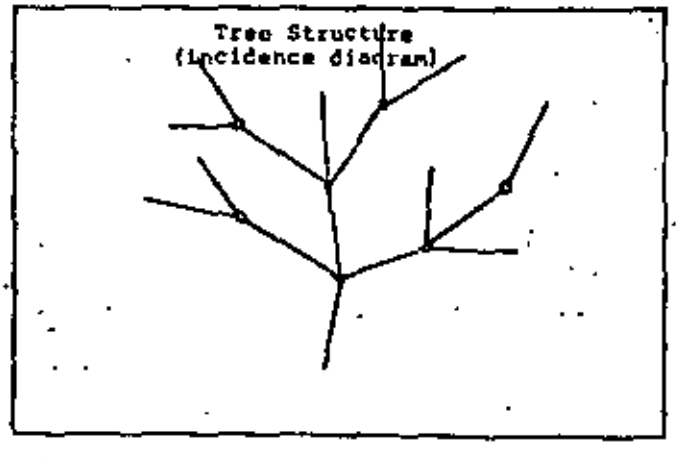


Figure 1

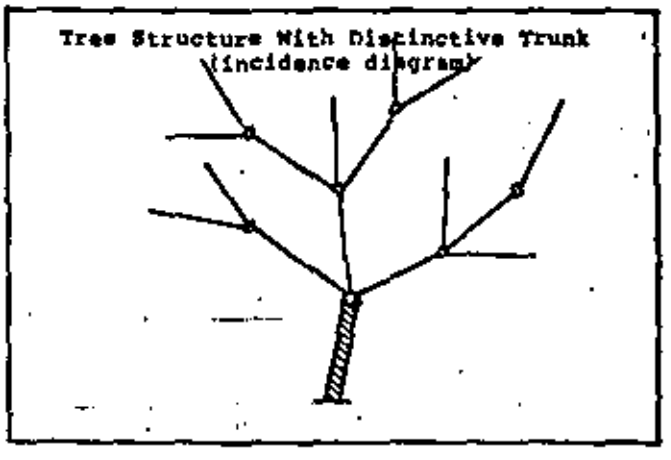


Figure 3

directories of directories of directories of. . . of named files. In corporate organizations we find companies which have departments, which have departments, etc. Figure 5 is a data structure diagram which illustrates this.

This example illustrates a specific need for this type of structure and supports the proposal. Some will argue that this is an incorrect approach to the fundamental corporate organization structure and that they are really hierarchies of different types of organizational-units. At one time, the General Electric Company had a well defined hierarchical organization structure which is illustrated in Figure 6.

If you were a "section" manager, you knew exactly where you were in the management hierarchy. This is an easier structure to handle manually than by computer, as people did not get quite as upset when someone thought it appropriate for a particular "unit" manager to report directly to a "section" or "department" manager. If this organization structure were declared to an I-D-S database system with the structure illustrated in Figure 6, then no "unit" could directly report to a "section" or "department", it would have to be assigned to a "subsection."

The proposal to support alternative owner record-types for a single set-type should be accepted because it is useful. It does not require that the database administrator use either the structure of Figure 5 (alternative owners) or Figure 6 (unique owners). It should be supported because it allows each administrator the choice.

The more general argument, for the support of alternative owner record-types for a set-type, also supports the need to retain the capability for multiple member record-types for a set-type. In an information system, real world entities are represented by records. These entities are classified by entity-type in order to facilitate the processing of data concerning them. Further, each entity-type may portray several concurrent roles or behavior patterns and sometimes these roles are shared by other distinctively different entity-types. For example, a person, or a company, or a governmental unit may serve in the role of "employer" of people, and as an "owner" of property. Within such a designated role, the record-types representing the entity-types should be capable of being the owner or member of a

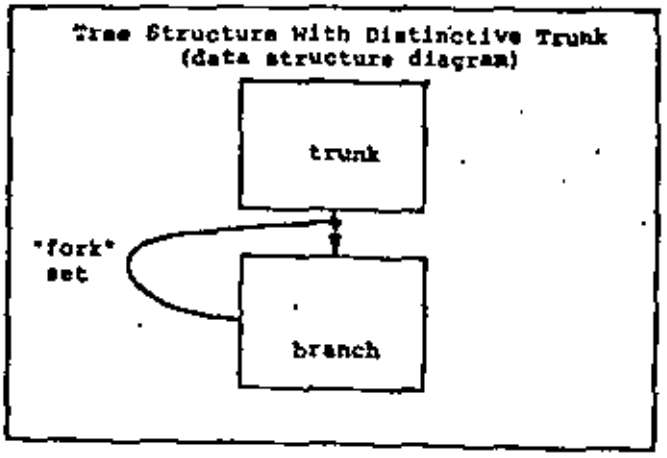


Figure 4

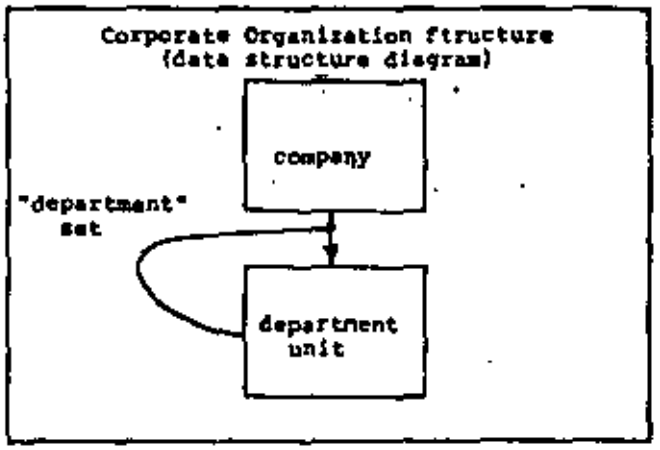


Figure 5

set-type which is role related, and be the holder of a field which is role related. Figure 7 is a data structure diagram which illustrates the employer role played by persons, companies and government units.

To model this structure, it is necessary to declare the "person" record-type, "company" record-type and "gov-

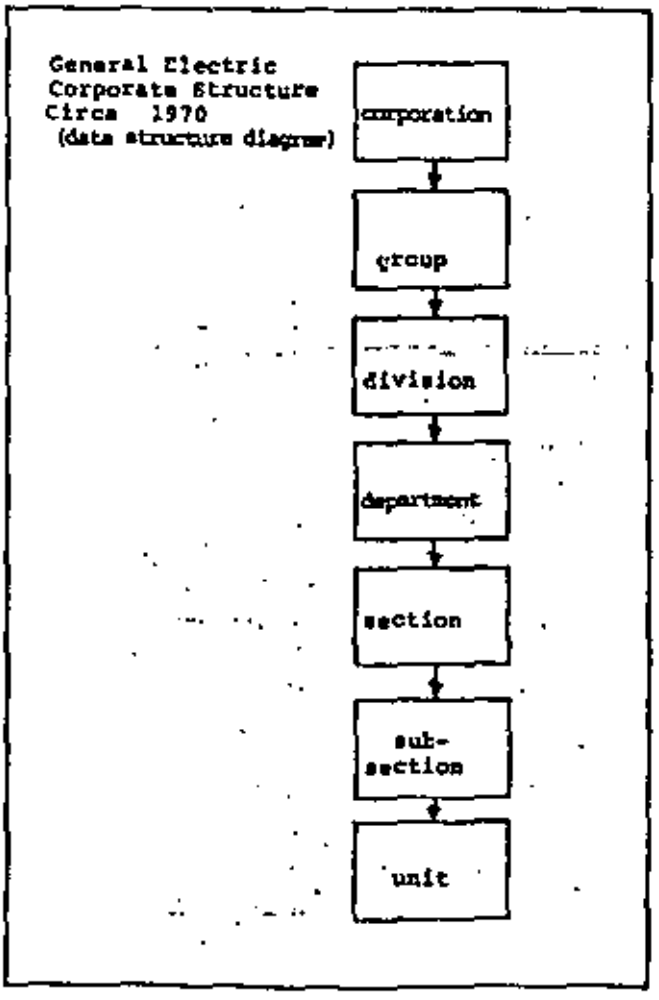


Figure 6

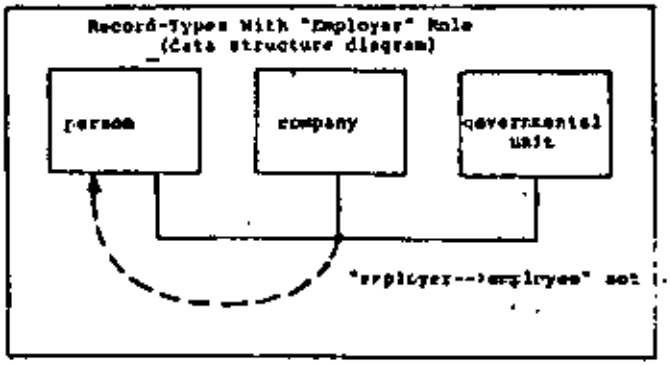


Figure 7

ernmental unit" record-type such that all are able to assume the role of owner of the "employer——→employee" set. It is necessary to declare the "person" record-type as a sometime set member, "sometime," since all persons are not necessarily employers.

In the case of a person who is self-employed, the same "person" record would be the owner and member of the same occurrence of the "employer——→employer" set. Both the alternative owner (prop. 3) and recursive set (prop. 1) proposals would need to be accepted to support this structure.

The reader should glance back to Figure 6, one of the possible means of implementing the organizational unit aspects of a corporate structure. Given this structure, now imagine how the organization-to-employee relationship would have to be handled. Each organizational record, from the "corporation" record through to the "unit" record, must be able to handle the role of employer. All seven of the organizational records need to be declared as alternative owner types to the "organization——→employee" set. Figure 8 illustrates this extension to Figure 6.

If one assumes that each of the units needs to have a manager, who is a person, then each of these seven organizational unit role record-types must also be declared as a member in the "managed" set. The support of the "managed" set gives an example of the usefulness of the ability to declared multiple record-types as members of the same set-type. (Proposal 2). Figure 9 illustrates the further extension of Figure 6 to include the "manager-——→organization" set.

RECORD-ROLE CONCEPT

For the theoreticians (and it is they who have largely argued for reducing the number of member record-types declarable for a set-type to one, and keeping the owner types declarable to one) the introduction of the "record-role" concept may be of great importance. This is because there will be no argument from the practitioners over having only one role declared as the owner of a set-type and only one role declared as the member of set-type if roles become declarable entities. Furthermore, the owner and the member declarations could be restricted to be different "roles."

In the work at Honeywell Information Systems on this subject, the word "record-role" has been used to characterize the role concept introduced above. The following definitions apply:

- A "record-occurrence" is the database representation of a real world entity.
- A "record-role" is a declaration of the a collection of the properties (fields and sets) which a record-occurrence may represent on behalf of one role of a real world entity.
- A "record-type" is a declaration of a collection of one or more record-roles which a record occurrence may represent, while roles are all played concurrently by a real world entity.
- A "record-class" is the collection of all record-occurrence of a particular record-type. A record-occurrence is always in only one record-class, defined by a record-type.
- A "role-class" is the collection of all record-occurrences of a particular record-role. A record-occurrence

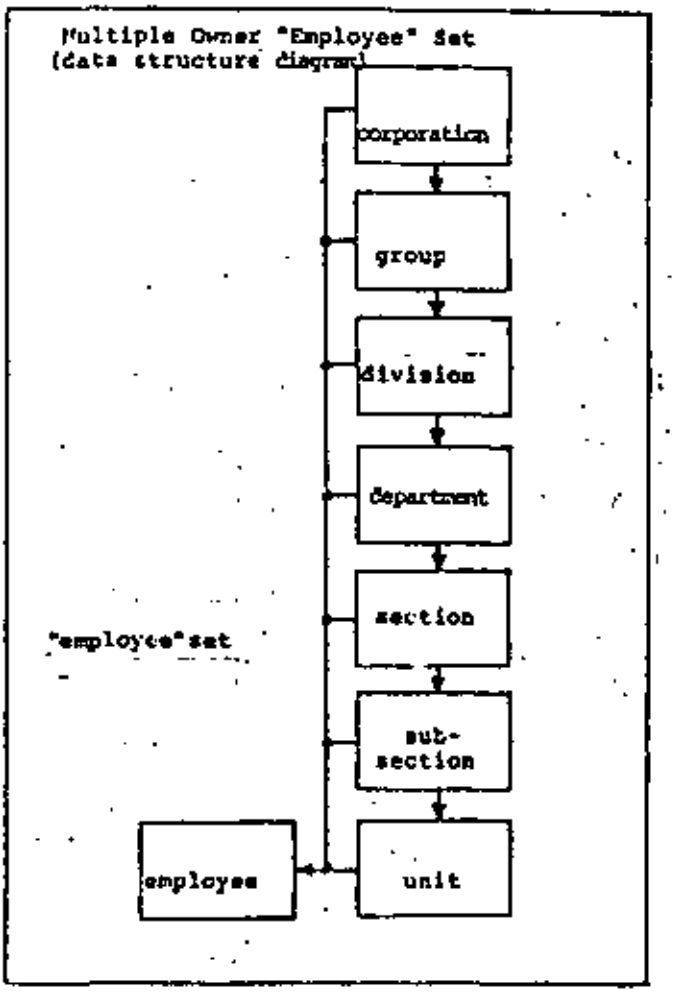


Figure 8

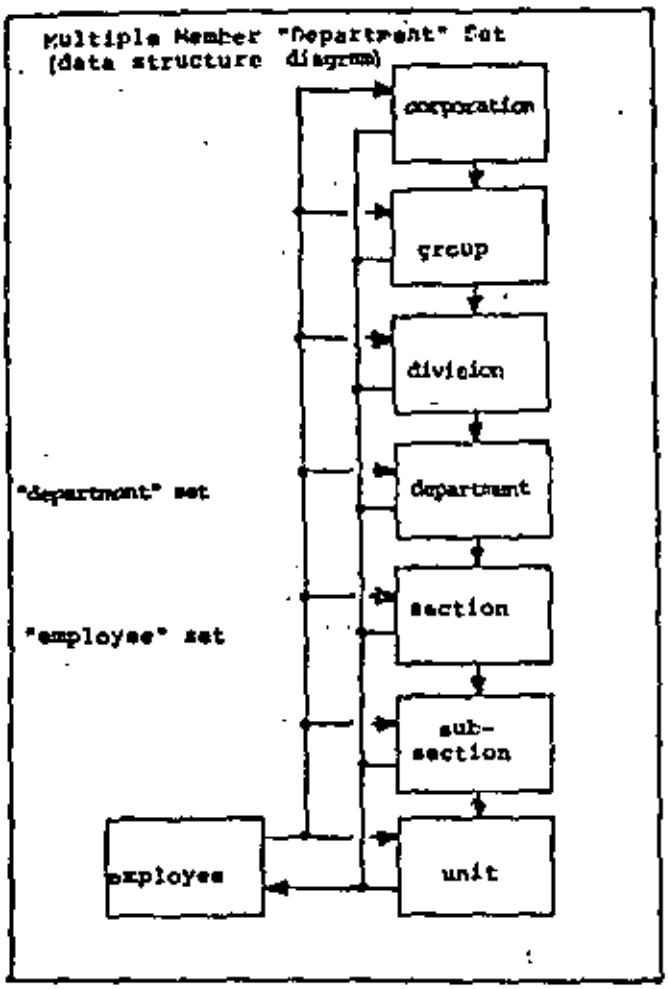


Figure 9

is in one or more many role-classes depending upon its record-type declaration.

- A "role-occurrence" is a subdivision of a record-occurrence which is the representation of that record-occurrence playing one role.

This distinction between record-role and record-type has not been made in existing database systems. While a record-type may have represented several record-roles, there was no mechanism of sharing the record-role declarations between two or more record-types. Thus the same fields and sets, relative to the role, had to be multiply declared, once for each record-type which played the role. This led to the requirement for multiple member declarations and alternate owner declarations for sets.

With the record-role concept, the declaration of fields, groups and sets are all associated with the record-role declaration. Field may be accessed using field-names which are qualified by record-role-name rather than record-type-names. Sets are ordered by role declared fields. Set owner selection is based upon role declared fields. Record occurrences of different record-types coexist within the same set type as owners or members when the record-types share the same record-role with the declared function.

In the paper "The Evolution of Data Structures,"[1] there was a sequence of data structure diagrams which were used to illustrate the progressive introduction of new meta objects and new inter-object relationships into data structuring capabilities. The first two of the following three data structure diagrams are reprinted from that article. Figure 10 illustrates the meta entities: record, field, group, owner, member and data-structure-set, where there may be an unlimited number of member record-types and owner record-types declarable for a set-type. The diagram of Figure 10 is the meta object structure which I recommend to DBTS and have recommended and used for a number of years.

Figure 11 is a simplification of Figure 10 where the restrictions of a one owner entity-role and a one member entity-role have been placed on the set-type. The set-owner and set-member meta entities have been merged with the set-type meta entities, as they exist on a 1:1:1 basis. This yields the restricted structural capability which was recommended by some of the attendees at the Namur Conference.

The data structure diagram of Figure 12 introduces the meta entity "record-role". In this structure, the "record-role" meta entity has displaced the "record-type" meta entity in its direct relationship to the set, group and field. The record-type concept is now at the side, associated by a declared relationship with one or more "record-roles".

"Record-role" declaration may be associated with one or more record-types. From the viewpoint of the set-type, there is only one "owner" record-role and only one "member" record-role. This fits more easily into the viewpoint of both the relational model and the Data Independent Access

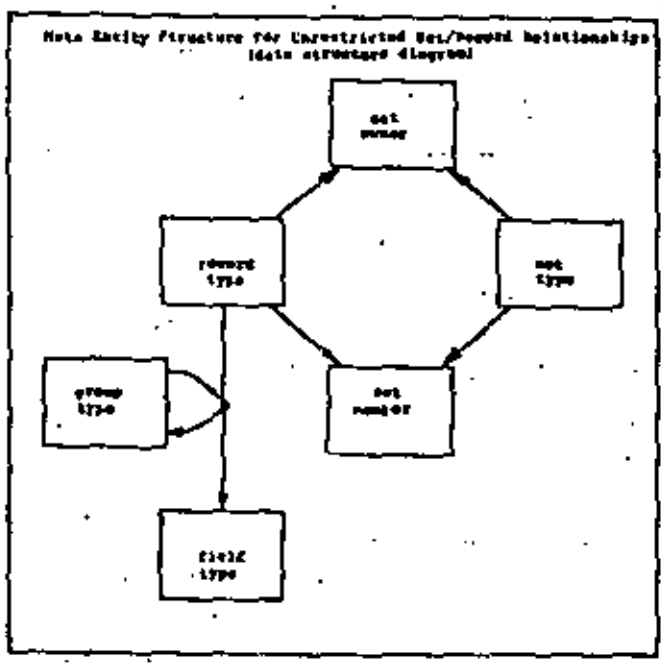


Figure 10

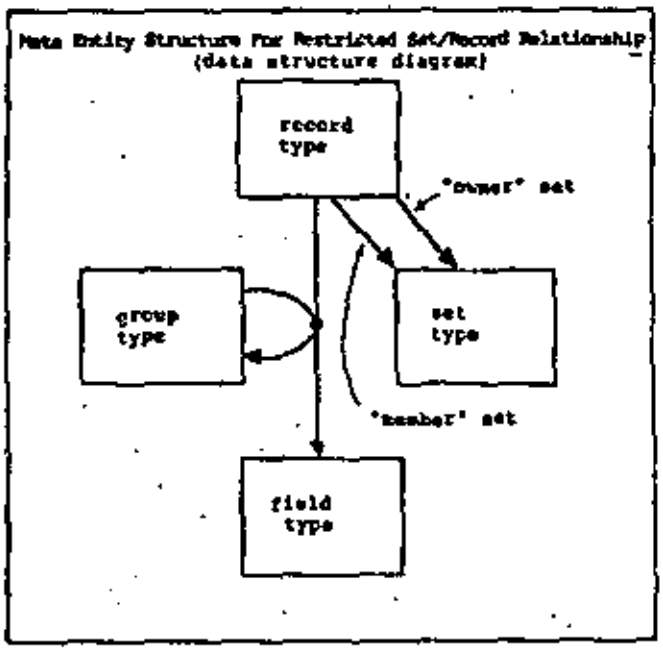


Figure 11

Model. However, the requirements of the real world which we wish to model can be satisfied as each real world entity can be recognized acting in one or more roles and its record-occurrences are combined with the declared role occurrences.

At this time I have no interest in trying to introduce the "record-role" concept into existing data description languages and data manipulation languages. Rather, I wish to provide the rationale, within these languages, for:

(1) recursive set-types,
(2) multiple member set-types, and
(3) multiple owner set-types

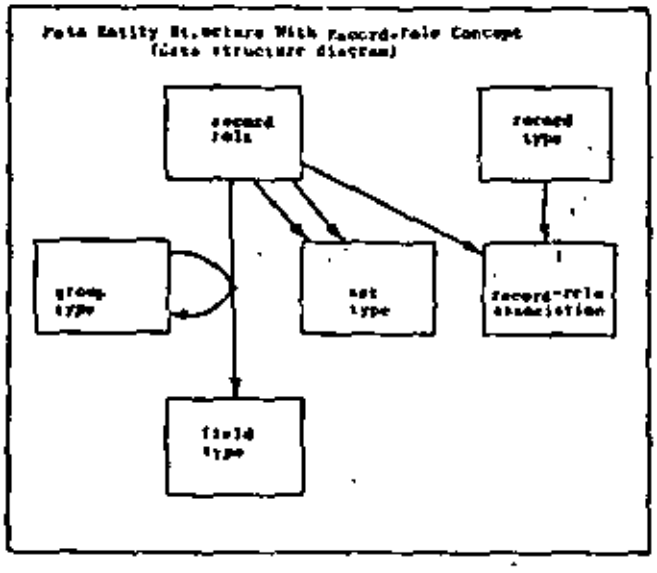


Figure 12

which provide an alternate means for achieving the objectives achieved by the record-role concept while remaining within the limitations of the presently available meta entities. However, the eventual introduction of the "record-role" may be the unifying factor that we seek.

The data structure diagramming technique has been extended to support the concept of record-role. Figure 13 is a redrawing of Figure 7 with focus on the record-roles of employer and employee.

They are illustrated by the two hexagons so designated. The record-types with which the roles are associated are designated by the background boxes. The employer role is played by the company, person and governmental-unit record-types. If classical data structure diagrams were thought to represent record-types and the relationships between them, then the new diagrams illustrate record-roles, their associations with record-types and their relationships with other record-roles. Record-roles are illustrated by hexagons and the background boxes name the record-types which play the role. Each record-type is considered to have one or more roles. In this example "person," "company" and "government-unit" are the record-types. "Employer" and "employee" are record-roles. A complete data structure diagram would show each record-type once as a box on top of a stack of hexagons. Each hexagon representing a record-role played by the record-type. It would also show each record-role, once as a hexagon at the top of a stack of boxes. Each record-type would appear once more as a background box behind each record-role it plays. Figure 13 is thus an incomplete data structure diagram as it shows the record-types playing each record role but does not graphically illustrate each record-

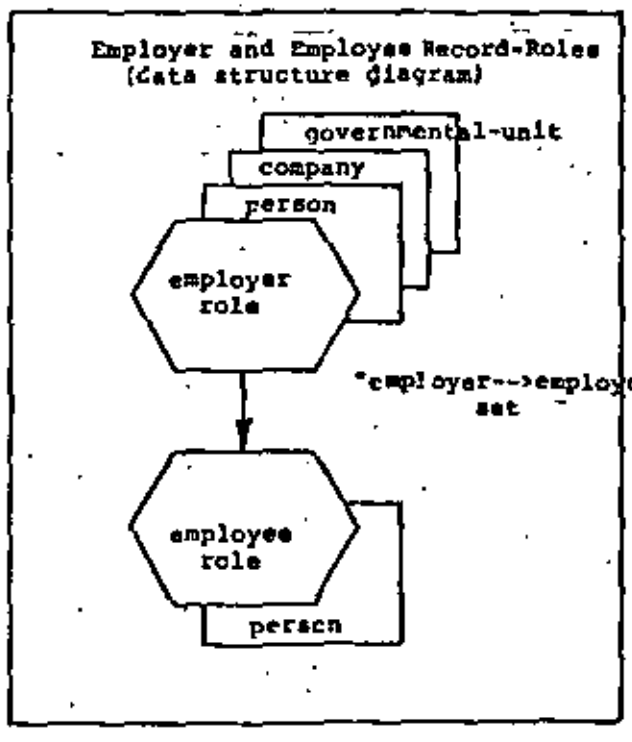


Figure 13

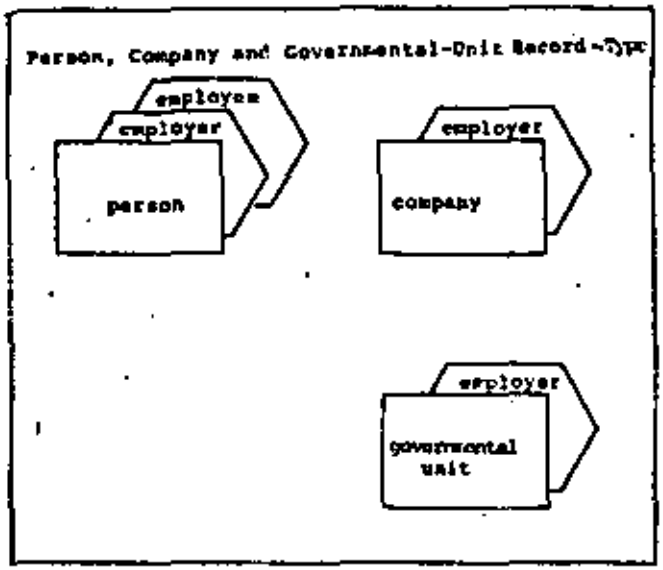


Figure 14

type with its record-roles. Figure 14 illustrates each record-type with the record-role that it plays. Thus most old data structure diagrams can be considered as being examples where the record had only one record-role. Thus no role factoring is necessary. If alternative owners or multiple members exist in those diagrams, then the record-roles for them has not yet been factored. The importance of the record-role concept to data structure diagrams may not be immediately obvious at the first comparison of Figures 7 and 13. However, consider the following analogy. If identical programming code appears in several parts of a computer program, it is common to factor this code out as subroutines or at least as macro procedures so that the documentation is more easily understood. The record-role concept is the data structure diagram equivalent of a subroutine call. The diagram illustrates the shared aspects of the record-role and also all the places it has been invoked. For data structure diagrams, representing complex organizations with many record-types, record-roles and their relationships, the record-role has proven to be extremely useful in simplifying the diagrams. They are much more readable. Of necessity, these diagrams are only effective after the new concept has been understood, used awhile and accepted.

SUMMARY

The data structure set is almost the only structural tool currently available to the database administrator to represent the relationships between entities in his enterprise. At this time when all of its usages are unknown, it seems desirable not to place any restrictions upon its application. Proposal 1, to permit recursive sets (where owner and member are of the same record class), is a proposal to remove a restriction. Proposal 2, to prohibit multiple member record-types, is a proposal to add a restriction. Proposal 3, to permit alternative owner declarations, is a proposal to remove a restriction. These facilities, within today's record-network model would provide a workable implementation of the role concept, an expression of the evident and important multiple behavior patterns which are characteristic of real world entities.

REFERENCES

1. Bachman, C. W. and S. B. Williams. "A General Purpose Programming System for Random Access Memories." *Proceeding AFIPS Conference Proceeding*. FJCC Volume 26 AFIPS Press Montvale N. J., 1964 pages 411–422.
2. "Integrated Data Store," *DPMA Quarterly*, January 1965.
3. "Software for Random Access Processing," *Datamation* April 1965, pages 36–41.
4. *CODASYL Data Description Language Journal of Development*, June 1973, (#13.6/22:113) Superintendent of Documents, U. S. Government Printing Office, Washington D. C. 20402.
5. *CODASYL COBOL Journal of Development*, January 1976, (110-GP-1D) Material Data Management Branch, Dept. of Supply and Service, 5th Floor, 88 Metcalfe Street, Ottawa, Ontario, Canada, K1A 0S5.
6. "Data Structure Diagrams, Data Base 1, 2," 1969, *Quarterly Newsletter of ACM SIGBD*, pages 4–10.
7. "The Evolution of Data Structures," *Proc NordDATA Conference*, August 1973, Copenhagen, Denmark, pages 1075–1093.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

DATA MANAGEMENT FOR CLINICAL RESEARCH

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

Data management for clinical research

by W. L. SIBLEY, M. D. HOPWOOD, G. F. GRONER, W. H. JOSEPHS and N. A. PALLEY
The RAND Corporation
Santa Monica, California

ABSTRACT

This paper discusses a prototype system intended for the personal use of physicians engaged in clinical research. In particular, the prototype is a highly integrated, interactive, minicomputer based data management and analysis system. The facilities offered by the system allow the clinical investigator to store, recall, analyze, and display his research data without resorting to computer programming. Modern data base techniques are available to the physician as aids in organizing, storing, and retrieving his data. The data base concepts are expressed in terms that are familiar to a clinical researcher.

PURPOSE

The purpose of this paper is to discuss a prototype system intended for the personal use of physicians engaged in clinical research. In particular, the prototype is a highly integrated, interactive, minicomputer based data management and analysis system.

THE CLINFO PROJECT

The CLINFO prototype data management and analysis system described in this paper has been developed as part of the CLINFO project, a scientific inquiry sponsored by the Division of Research Resources (DRR) of the National Institutes of Health (NIH). The goals of the project are to identify and characterize the information analytic tasks and the information flows in clinical research, and to develop methods for facilitating these tasks and flows. The project is being conducted by clinical investigators at the Baylor College of Medicine, the University of Washington, the University of Oklahoma, and Vanderbilt University; by information scientists at the Rand Corporation; and by staff members of the DRR.

We have thus far (1) interviewed clinical investigators both formally and informally, (2) characterized their information processing needs, (3) identified data management and analysis as major problems, (4) examined existing computer systems aimed at satisfying those needs, (5) designed, built, and installed three copies of a prototype system, and (6) started the analysis of instrumentation data and user interviews to help us evaluate the system.

References 1, 2, 3, and 4 discuss various aspects of the foregoing points.

CLINICAL STUDY DATA VOLUMES AND ACTIVITIES

A General Clinical Research Center (GCRC) provides facilities which clinical research personnel can use in conducting a variety of medical studies. Each study tends to be separate and distinct from other studies and provides a natural focal point for data collection and analysis. A large study involves approximately 75 subjects (patients) and a total collection of approximately 67,000 data values.[1]

Generally, a clinical study involves the following activities:

- During the study design the investigator decides what data are to be collected, at what rate, and in what volume.
- As the study progresses, data are collected and recorded in a central data file. Considerable care is taken to ensure the validity of those data.
- As the study progresses, the investigator reviews and summarizes the data. Part of the process of summarization involves transcribing the data into a form suitable for statistical analysis as well as the preparation of plots, graphs, and reports.

The CLINFO prototype is designed to accommodate 25 such studies.

SYSTEM OVERVIEW

The prototype is implemented on a commercial minicomputer (Data General Eclipse S/200) using multi-user extended BASIC.[5] Each study has its own portion of a 25 million character disk. When an investigator (or a member of his staff) uses the system, he interacts with the CLINFO software (primarily by responding to prompts) and is prevented from using BASIC procedures to affect his data.

Figure 1 illustrates the main features of the CLINFO

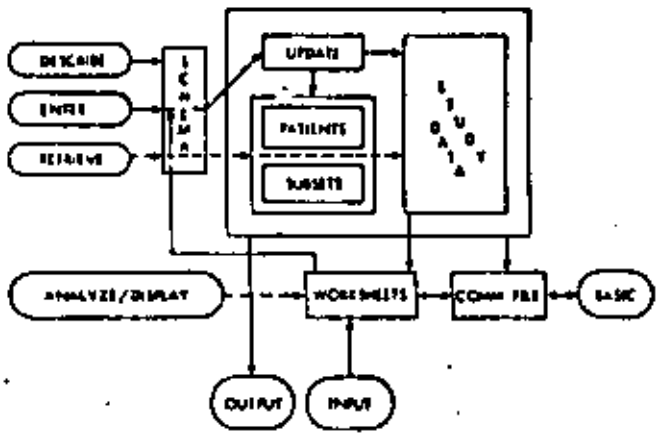


Figure 1—Command(···›) and data(——›) flow in the CLINFO prototype for a single protocol

prototype. The ellipses encompass the three general activities which were mentioned above. The rectangles indicate the various ways in which data (or descriptions of data) are stored in the prototype. The diagram encompasses a single study (or research protocol). This paper is concerned primarily with the activities labeled "DESCRIBE," "ENTER" and "RETRIEVE" and the data contained in the "SCHEMA," "UPDATE," "PATIENTS," "SUBSETS," and the "STUDY DATA" structures. The "ANALYZE/DISPLAY," "INPUT," "OUTPUT," and "BASIC" activities and their associated structures will be described, but in less detail.

THE DATA BASE

The organization of a CLINFO study data base is predicated on three observations:

- The fundamental unit of analysis is usually the patient.
- Data are collected about that patient over time.
- The data have natural groupings in time (e.g., the vital signs of a patient are sampled at essentially the same time).

The time-oriented nature of clinical research data has been discussed by Fries.[6] Dr. Fries' ideas have been further reinforced by our observations of the current manual techniques for recording that data. For example, data are recorded in patient flowsheets (one for each patient) which are two-dimensional arrays with the rows representing the data items to be measured and the columns containing the values of an item for different times of measurement. We have further observed that some of the rows can be grouped naturally into related items (examples of such groups are vital signs, blood serum tests, urine analyses, etc.). However, the groupings differ widely from study to study. The CLINFO term for a grouping is "PANEL." The concept parallels records or groups in the CODASYL Data Base Task Group (DBTG) report.[7]

PATIENTS/SUBSETS

The key structure is the PATIENT file which contains an eight character abbreviation for each patient (assigned by the clinician when the patient is added to the study) and an internal numeric key (assigned by the system). This file can contain up to 392 different entries. The entries are ordered alphabetically on the patient abbreviation to allow for efficient searching.

A subset is a named file (e.g., LOWBLOOD) with the same structure as the PATIENT file but containing patients with particular properties. No particular patient ordering is maintained for subsets. There can be as many subsets as space allows.

The numeric key associated with each patient references a block of data in a PATIENT DIRECTORY (not shown). That block contains some redundant identification data and two tables. The first table contains pointers to the patient's set of panels (see above) in the STUDY DATA file and counts of the numbers of instances of the panels. The second contains data concerning the state of the various EVENTS (to be described later) for this patient.

DESCRIBE/SCHEMA

The SCHEMA[3] provides the means for adapting the CLINFO data base structure to the needs of a particular investigator. It is a description of the panels and their contents. The schema exists in two forms: the external textual form and the internal compiled form. The former is for human interaction, the latter is for computer processing. The DESCRIBE activity is in essence an interactive editor which allows the clinician to create, edit, and compile his schema.

The schema describes three main entities: ITEMS of data, PANELS of items, and EVENTS triggered by the occurrence of certain values for particular items.

PANELS have eight character names, and references to them by name imply reference to all of their items. There are two types of panels, numeric and textual. The type of the panel determines the types of the data items included in that panel. Each panel can contain at most 30 items, all of which are either 32-bit floating decimal numbers, or character strings at most 70 characters in length. A sample panel entry in the schema might be:

```
PANEL  1 hist   , patient ID & history,numeric;
```

Reading from left to right, this is the first panel (PANEL 1), its name is "hist ", it contains "patient ID & history,", and its items are all numeric.

ITEMS also have eight character names which are used to reference the data associated with an item in a wide variety of circumstances. Again, an item may either be numeric or textual. In addition, there are parts of the item description that serve to screen, validate, and possibly encode the value associated with that item. For example:

```
ITEM 9 HT ,Height   ,inches   ,num,range,(55,80),no;
```

Reading from left to right, this is the ninth item in the schema (ITEM 9), its name is "HT ", its values represent height in inches, its values are numeric and must lie in the range 55 to 80 inches inclusive, and its value need not be considered confidential ("no"). "Height" and "inches" are purely descriptive and do not imply automatic units checking by the system.

The sequence "num,range,(55,80)" illustrates one option available for data screening. Other options for items in numeric panels are:

- date,range(01031976,12251980)
  the item value is a date in the indicated range (1/3/1976 to 12/25/1980 inclusive).
- time,range(1300,1330)
  the item value is a time in the indicated range (13:00 to 13:30 inclusive).
- num,check(4),(7,10,6,5)
  the item value is numeric and one of the listed 4 values.
- char,code(3),(yes ,no ,unk)
  the item value is externally one of the listed 3 character strings and is carried internally as one less than the ordinal position of the string in the list (i.e., 0, 1, or 2).
- the data types date, time, and num may require no validating, e.g., num,none.

The textual panels and items are of the form:

```
PANEL  2,demo   ,demographic data,text;
ITEM   20,name  ,patient name,text(30);
```

The item "name" will then be allocated 30 characters in each occurrence of a "demo" panel.

EVENTS are time markers maintained by the CLINFO prototype system to aid the user in the retrieval of his data. It is common in clinical research to relate the response of a patient to a procedure by referencing some events which marks the application of that procedure. For example, blood sugar levels are measured at regular intervals after the ingestion of glucose. Events provide the means for dealing with relative time. A sample event is:

```
EVENT 3 test ,glucose ingest,glue ,FIRST;
```

Reading from left to right, this is the third event in the schema, its name is "test," it relates to glucose ingestion, and it records the time at which the item "glue" FIRST takes on a value. Time is taken here to mean a combination of date and time.

Events may be triggered by the "FIRST," "LAST," "MAX," and "MIN" values of its governing item. In addition, an event may be triggered at the first or last time the item takes on a particular value.

When the clinician is satisfied with his description of the data he intends to collect, he compiles the schema (the system provides appropriate feedback about errors in the compilation). He then requests that disk space be allocated to accommodate his study. He assists the allocation by estimating the number of patients he expects to include in his study and an approximate number of occurrences of each panel type for the average patient.

He is then ready to start entering patients and their data into the data base.

ENTER

The main function of the ENTER activity is to process data (entered interactively) in the light of the data screening specifications contained in the schema. The data that are accepted are not passed directly to the STUDY DATA file, but are recorded in an UPDATE file whose contents are merged with the STUDY DATA file at some convenient time. This buffering of the input data allows a simple structure for the STUDY DATA file, reduces the exposure of the system to computer malfunctions, and provides a means of reviewing the input data before it is actually recorded in the STUDY DATA file.

Again, a CLINFO data base is patient- and time-oriented. In order to enter data, a "context" must be established for that data. That is, the data must

- Belong to a patient already in the PATIENT file.
- Have an associated date and time of day at which the data are assumed to have been sampled.

After a context has been established, (by interactive prompts and responses) the clinician may then enter values for items by either

- Typing the name of an item followed by its value or
- Typing the name of a panel, whereupon the system prompts for each of the items in that panel.

The context stays in force for all data entered subsequently until the clinician desires to change it. The context may be changed at any time by typing "patient:," "date:," or "time:" followed by the corresponding value.

As each value (including patient abbreviations, dates, times, and panel and item names) is entered, that value is checked for reasonableness. The patient abbreviation must be in the PATIENT file; dates and times must have the correct format; panel and item names must exist in the schema; the value for an item must satisfy the criteria in the schema. Incorrect values are refused and re-prompted for by the system. An exception is the case of an out-of-range numeric value which may be forced upon the system by entering the initials of the data enterer. Incorrect but reasonable values that have been accepted by the system may be corrected by re-typing them (perhaps after re-establishing the correct context).

ENTER has three additional functions:

- Adding patient abbreviations to the PATIENT file.
- Reviewing data in the UPDATE file.
- Copying and screening data from a worksheet into the UPDATE file.

As an alternative to item-by-item data entry, an entire two-dimensional array (i.e., a "worksheet") of data may be entered. A worksheet is a two-dimensional array of numeric entries extended to include 8-character labels for the rows and columns as well as a worksheet name, title, date of creation and date of last modification. A worksheet is stored by the system as a file with the same name as that of the worksheet. Worksheets provide the data organization required by the ANALYZE/DISPLAY activities. The row and column labels can be used to reference patients, relative times, and items. These labels are used to direct the data entry process. Data entry via worksheets provides the facility for entering "derived" data (i.e., data produced as the result of the analysis of raw data) into the STUDY DATA file. It also provides another data entry route that may be more convenient in some circumstances.

As indicated above, the clinician can interactively review the contents of the UPDATE file and edit it to the extent of deleting blocks of data.

When he is satisfied that the data are correct, the clinician can request that the UPDATE file be merged with the data already in the STUDY DATA file. The newly entered data are organized automatically into panels, those panels are marked with the internal patient identifier, the panel number, the date and time of sample, the date and time of data entry, and the initials of the data enterer, and the results are merged by date and time of sample. The merge process checks for existing instances of panels with the same date and time as those being merged and for item values previously entered in those instances.

The investigator is now in a position to retrieve his data from the STUDY DATA file and to format it in a variety of ways: into worksheets, as a display at the terminal, or as a file to be passed to an externally generated applications package (Such a file is a "communication" file in CLINFO terminology.).

RETRIEVE

The data in the STUDY DATA file may be viewed as values associated with points in a three dimensional space. Figure 2 illustrates that space. The axes are labeled "PATIENT", "TIME," and "ITEM" to reflect the parameters that uniquely identify a data value.

The three dimensional space is sparsely populated with data points, especially if the TIME axis is taken to represent absolute dates and times. Moreover, not all data are collected about all patients for all of the expected times.

It is the intent of the RETRIEVE activity to identify a planar slice of the data space and to abstract part of that slice into a two-dimensional form (i.e., a worksheet). Selecting a slice as illustrated in Figure 2 results in what was referred to above as a flowsheet for that patient. A slice parallel to the PATIENT-ITEM plane produces an array of items for a group of patients at a particular time. That time may be either an absolute date and time, or it can be a time relative to an event (e.g., two hours after glucose ingestion). Finally, a slice parallel to the PATIENT-TIME plane

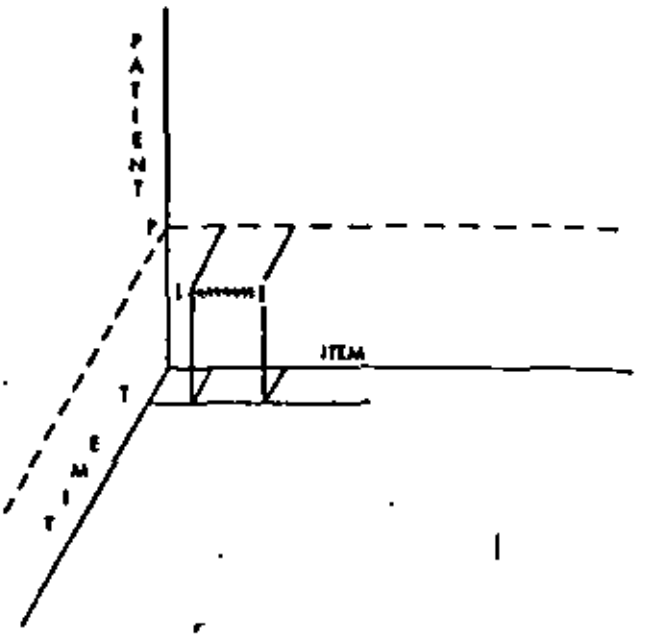


Figure 2—A three-dimensional representation of a time-oriented data file. I is a panel of items for patient P at time T

results in a worksheet containing the time history of a particular item for a group of patients (again either absolute or relative time).

The clinician constructs his retrieval request interactively under the control of the schema. In particular, coded items (e.g., sex may be either "male" or "female", but is encoded as either 0 or 1) are referred to in terms of the external form ("male," "female") while the system deals with them as numeric values (0 or 1). The values of item play a role if the clinician wishes to restrict the panels being retrieved by specifying conditions that the values must satisfy.

When he completes the specification, the system surveys the appropriate parts of the STUDY DATA file and produces a worksheet with the rows and columns appropriately labeled.

The retrievals discussed above are limited to numeric (or coded) items. However, textual panels may be retrieved and displayed at the terminal or passed in their entirety to a communication file. This facility allows the clinician to review his notes about a patient or to pass demographic data to a BASIC program that, for example, produces mailing lists.

For those retrievals in which a list of patient abbreviations is appropriate (e.g., a slice parallel to the PATIENT-ITEM plane), the investigator may provide that list by giving the name of a SUBSET.

SUBSETS

As mentioned above, subsets are files containing patient abbreviations and numeric keys. The files have the same structure as the PATIENT file.

Subsets are created in a variety of ways:

- By listing patient abbreviations.
- By the usual set operations (e.g., intersection and union) among existing subsets.
- By requiring that a patient's data in the STUDY DATA file satisfy a set of conditions.
- By requiring that a patient's data in a worksheet (e.g., the values in a row labeled with the patient's abbreviation) satisfy a set of conditions.

The sets of conditions mentioned in the last two members of the above list are made up of lists of either conjunctions or disjunctions (but not both) of range restrictions on the values of items. A sample set of conditions is:

If	30 ≤	age	≤ 65
and if	120 ≤	systol	≤ 140
and if		sex	= male

The last line illustrates the use of a coded item which takes on one of a set of discrete values.

In addition to the conditions placed on item values, the time at which a sample was taken can play a part in the selection. For example, the above conditions might be required to hold in the third hour after the ingestion of glucose.

In any case, the usual result of a retrieval using either a subset or all of the PATIENT file is a worksheet.

ANALYZE/DISPLAY

The CLINFO approach to data analysis is to abstract the desired data from the STUDY DATA file (which can cause very intensive accessing of the disk storage) into a worksheet which fits into the user's working space in main memory. The worksheet can then be manipulated efficiently, especially since its contents need not be re-abstracted from the STUDY DATA file each time a new question is asked of those contents.

A sample worksheet has the format illustrated in Figure 3.

WORKSHEET CLINICAL
TITLE Clinical Characteristics of Diabetic Patients
CREATED 12/13/75 MODIFIED 12/13/75
OF ROWS 7
OF COLS 4

ROWS/COLS	1	2	3	4
LABELS	age	sex	%ideal wt	dur diab
1 case 1	27	male	100	12
2 case 2	24	female	98	11
3 case 3	32	male	98	13
4 case 4	34	male	94	20
5 case 5	21	male	96	9
6 case 6	23	male	...	...
7 case 7	28	female	99	10

(Note: . . . indicates missing values)

Figure 3—A sample worksheet

A worksheet can hold approximately 2200 entries and its dimensions can vary within that restriction. In the case above, the row labels are patient abbreviations and the column labels are item names. These labels and the contents of the worksheet would ordinarily be supplied by the retrieval process. In addition, Figure 1 indicates an INPUT facility that allows the direct creation of worksheets, the labelling of their rows and columns, and the entry of data into those rows and columns. This facility allows the investigator to use CLINFO's analysis and display features independently of the STUDY DATA file.

CLINFO maintains a "current" worksheet so that the same worksheet can be used in a variety of situations without the user respecifying it as the worksheet of interest. Most of the worksheet analysis and display functions apply to the current worksheet. The main worksheet manipulation facilities are:

- Select a worksheet as the current one by typing its name. If there is no worksheet by that name, create one as specified by the user.
- Display a worksheet at the terminal (see Figure 3).
- Label worksheet rows or columns.
- Enter data into a worksheet by row or by column or by individual cell.
- Discard (destroy) a worksheet.
- Sort a worksheet by rows or columns.
- Edit a worksheet by adding, deleting, or moving rows or columns.
- Copy a sub-array of a worksheet, with labels, into the same or another worksheet.
- Print a worksheet on the system printer.
- Display a list of all the worksheets in this study.

Worksheets created by the retrieval process or by direct means can be used by the analysis facilities. In general, the analyses can be made to apply to sub-arrays within the body of a worksheet. The results of some analyses may be stored in worksheets to minimize transcription and to make those results available for further analysis.

The CLINFO analysis facilities include:

- Descriptive Statistics (means, standard deviations, etc.).
- T Test.
- Chi Square Test.
- Linear Regression (simple and multiple).
- Analysis of Variance.
- Cross Tabs.
- Scatter Plots and Bar Charts.
- Histograms.
- Frequency Distributions.
- Normality Test.
- Non-parametric Paired Tests.

In addition, special calculations may be performed on a worksheet, and the result of these calculations are stored in the worksheet. This facility replaces in part the need for special purpose computer programming. These calculations

are:

- Generate values in a row (or column) as the result of evaluating a single algebraic-like expression involving other rows (or columns) as variables.
- Generate truth values (0, 1) on the same basis.
- Perform special calculations such as cumulative sum, time differences, etc.

SUMMARY

The foregoing discussion does not describe the properties and facilities of the CLINFO prototype in their entirety. The intent is rather to outline those properties and facilities and to emphasize the idea of a personal, integrated, highly interactive system placed in the hands of personnel whose primary interest is to use the system as a tool.

At least one CLINFO prototype has been in daily use by medical personnel since January of 1976. The acceptance of the system seems to be good and it is being used in productive ways. Moreover it is being used personally by senior medical staff, that is, the research staff for which it was designed.

Our experience to date indicates that modern data base techniques (e.g. the SCHEMA), when expressed in the user's own terminology, are readily understood, accepted, and used by those personnel. The use of terms such as "PATIENT", "PANEL", and "WORKSHEET" and the time orientation of the STUDY DATA file are not accidental. They play an important role in making the computer scientist's techniques understandable by and useful to the clinical researcher.

Another point of interest is the utility of two distinct data structures, the STUDY DATA file (and its associated files) and worksheets. The STUDY DATA file acts as a pool of data whose contents can be abstracted and arranged into a form more suitable for viewing and analyzing (i.e., a worksheet). In addition, worksheets serve to store data and to communicate them between steps in the analysis process.

ACKNOWLEDGMENTS

Although the authors accept the responsibility for the words in this paper, a much wider group is responsible for the content those words express. We would like to express our appreciation to our co-contractors Howard K. Thompson, Jr., M.D., at the Baylor College of Medicine, T. Graham Christopher, M.D., at the University of Washington, and Arthur W. Nunnery, M.D., at the University of Oklahoma, our project officer, William R. Baker, Jr., Ph.D., of the NIH, and to our colleague Thomas L. Lincoln, M.D., of the Rand Corporation. They have all contributed significantly to the design and continuing development of the CLINFO prototype. This research was supported by Contract No. NO1-RR-5-2111 from the Division of Research Resources, National Institutes of Health.

REFERENCES

1. Palley, N. A., and G. F. Groner, "Information Processing Needs and Practices of Clinical Investigators—Survey Results," *AFIPS Conference Proceedings* (1975), Vol. 44, AFIPS Press, Montvale, New Jersey, 1975, pp. 717-723.
2. Groner, G. F., N. A. Palley and N. Z. Shapiro, "A Structural Characterization of Clinical Research Participants and Their Activities," R-1540-NIH, The Rand Corporation, January 1975.
3. Groner, G. F., M. D. Hopwood, N. A. Palley, N. Z. Shapiro, and W. L. Sibley, "A Plan for the Development and Evaluation of a Data Management and Analysis System for Clinical Investigators," R-1543-NIH, The Rand Corporation, August 1974.
4. Sibley, W. L., M. D. Hopwood, G. F. Groner, and N. A. Palley, "A Prototype Data Management and Analysis System for Clinical Investigators: An Initial Functional Description," R-1621-NIH, The Rand Corporation, August 1974.
5. Data General Corporation, *Extended BASIC User's Manual*, publication 093-000085-4, September 1973.
6. Fries, James F., "Time-Oriented Patient Records and A Computer Databank," *The Journal of the American Medical Association*, Vol. 222, December 18, 1972.
7. CODASYL Data Base Task Group (DBTG) Report, April 1971. Available from the Association for Computing Machinery.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

A METHODOLOGY FOR MULTI-CRITERIA INFORMATION SYSTEM DESIGN

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

A methodology for multi-criteria information system design*

by JOHN S. CHANDLER and THOMAS G. DeLUTIS
The Ohio State University
Columbus, Ohio

ABSTRACT

The design dilemma faced by the designer is to satisfy a set of conflicting user demands and resolve a set of conflicting resource requirements concurrently. In light of the complexity of modern systems, it is assumed that good system design need only produce satisfactory performance for both criteria. Current evaluative techniques, however, concentrate on either the user criterion or the system criterion aspect of the total design problem, but not both. A methodology has been developed that establishes a formal liaison between the evaluation of user goals as a function of system activity and the evaluation of resource utilization as a function of user demand, thereby creating a design/evaluation process that encompasses both criteria. The methodology employs three stages in an iterative manner to produce a "satisfactory" design. The IPSS simulator models and measures system activity, multiple goal programming evaluates both user and system goals, and heuristic procedures determine design modifications to improve performance. A functional description of the methodology and an example of its use will be presented.

INTRODUCTION

In March of 1973, ACM and NBS sponsored a Workshop** on computer performance evaluation. One of the major results of that Workshop was a consensus that there have been two separate approaches to the evaluation of information systems performance—one which focuses on the computer system domain and the other whose attention is directed at the application system (user) domain. Each have their own goals and measures: the computer system domain measures are based on resource queueing and utilization statistics and the user domain is evaluated through the performance of requested services. Measures such as throughput and response time are common for the latter. The Workshop also concluded that any performance analyses "should recognize both the costs of a computer installation and the needs of users for service."

The complexity of the design problem for modern computer based information systems has increased significantly over its predecessors due to:

a. the servicing of an expanding range of user or uses with corresponding diverse performance goals and resource requirements, and
b. the dynamic and unpredictable behavior of the system as a function of design decisions and load mix.

Thus, it is quite possible to improve the performance of the system with respect to one or more users at the expense of others. Likewise, because system resources are used by different users, improving the performance characteristics of one or more resources for the benefit of specific users may have an overall detrimental effect on performance. The problem presented to the designer is to configure a system which satisfies the user criterion while achieving system resource related performance criteria.

A computer based information processing system can be viewed as a symbiotic relationship between the system's users and its hardware, software and data resources. Ideally, the system will perform "optimally" when it achieves its user oriented objectives within a minimum cost system. However, optimal solutions are seldom achieved when systems are complex, ill defined or constrained for reasons outside the control of the designer, and thus, the designer usually settles for a satisfactorily behaving system. Hopefully, systematic procedures are employed to achieve system configurations which concurrently meet the user objectives while obtaining efficient utilization of its resources. Current evaluative technologies focus on only one criterion in the system design equation, either the user or the system's resource performance. The ability to simultaneously ascertain the impact of resource performance on user goals or vice versa is not readily achievable through these methodologies. The purpose of this paper is to describe a methodology which establishes a formal liaison between the evaluation of user goals as a function of system behavior

* This research is being conducted with the support of The National Science Foundation, Grant No. SIS75-21648.

** This was one in a series of Workshops sponsored jointly by ACM and NBS to examine the major issues involving computers. Performance evaluation was chosen as the topic of this Workshop because of its significant impact on computer usage. A summary of the conclusions appears in Reference 1.

and the analysis of resource performance as a function of user activity.

User oriented analyses with objective functions based on response time, throughput, and cost have been (and are continuing to be) reported in the literature. Most frequently, analytic approaches use queueing models as their basis (References 2-4 are representative of this type of analysis). Due to the necessity to maintain tractable models, many simplifications are required for a model's analytical solution. Simulation models have also been applied to user oriented analysis.[5,6] Unfortunately, these models yield only average and/or aggregate measures of system response. As a result of these simplifications, the analyses produced by both of the approaches fail in many cases to identify the relationship between users and resources. Therefore, they are suspect when used to predict the impact on system performances of modifying the current environment.

Alternatively, performance analyses can be made from the system's standpoint, treating the user and his goals in the aggregate. The most common approach is a subsystem study, where a particular part of the information system complex is isolated, with the subsystem user(s) represented by a stochastic generator, both analytic and simulative. The most emphasized areas of research has been the I/O subsystem[7-10] and CPU utilization.[11-13] The problem with this level of evaluation is that, although providing valuable local intuitive insight, these models rarely relate to the ultimate information system user, and, therefore, do not provide realistic insight into global performance.

Examinations of complete systems have also been made. Exhaustive hardware/software measurements have been analyzed by Gonzales and Cantrell[14-16] while simulation models, including an aggregate user component, have been built by Reeves and Pooch, Norland, and Lum.[10-18] Although results of the evaluations include resource utilization statistics and user oriented measures such as response time, there is little attempt in these models to relate particular resource usage to the effort on user goal attainment. (Two exceptions are Lindsay's study of the KRONOS system[19] and Hull's data base investigations.[20]) But from practical experience it is evident that there is indeed a relationship between user goals and resource usage. In fact, Buzen[21] has recently proposed some fundamental laws for computer performance which relate resource activity to global system/user measures such as response time and throughput.

It is assumed that the objective of good system design is to satisfy both performance related criteria. However, in light of the complexity of modern systems, many design decisions tend to be made without proper supportive evidence on performance. The crux of the problem is to establish a causal relationship between user goal attainment and system resource expenditures. The methodology to be discussed has been designed to establish such a liaison and will be shown to allow for the collection of heretofore hard to obtain evaluative information. The methodology measures the impact of individual user classes on internal system performance and identifies system bottlenecks which inhibit

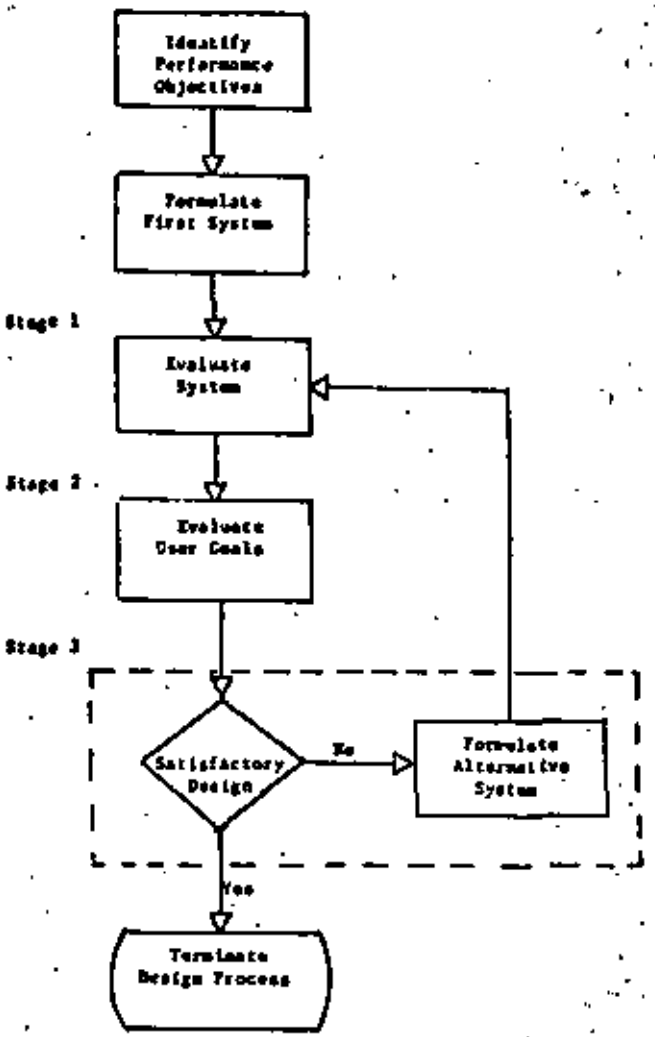


Figure 1—Stages in system design process

the attainment of user goals. This is achieved by maintaining resource utilization statistics on a user class basis. This methodology presents an evaluative framework which is capable of eliminating many of the numerous nonsatisfactory designs by directing the designer to the most advantageous ones. This methodology is an iterative one with each iteration involving three separate but integrated stages. Figure 1 illustrates the activities for an iteration. Briefly the responsibilities for each stage shown in this figure are:

Stage 1: System Evaluation

This stage is responsible for evaluating the behavior of a specific information system model. It does this by associating the hardware, software and data activities belonging to a specific design with the system's user activities. The outputs of Stage 1 are performance statistics for the resources in the aggregate and for their behavior with respect to identified users (or uses). To perform this function, the IPSS Simulator is employed.†

† IPSS is a special purpose discrete event simulator whose development was conducted with the support of the National Science Foundation, Grant No. GJ-36622.

Stage 2: User Goal Evaluation

Stage 2 has two purposes. The first is to ascertain whether the user goals are being either over or under achieved. The second purpose is to determine the "best" set of guidelines for altering the *current* system configuration in order to obtain the user goals with minimum penalty for either under or over achievement. Multiple goal programming is used for this purpose. As will be seen, "best" is a function of the assigned penalty coefficients in the goal programming objective function.

Stage 3: Design Evaluation

Stage 3 has two functions. The first is to ascertain whether or not the current design's performance is satisfactory with respect to both the user criteria and the system criteria. If the design is not satisfactory, then this stage's second goal is to define a new system based upon the current design, prior alterations, and the results of the Stage 1 and Stage 2 analyses. Heuristic procedures are currently employed for Stage 3.

The focus of this paper is on the Stage 2 formulation and its formal liaison to Stage 1. The paper also identifies the unique features of IPSS which permit this multi-stage multi-criteria methodology to be achieved. The paper concludes with a discussion of the use of the Stage 1 and Stage 2 results in the Stage 3 heuristics.

EVALUATIVE REQUIREMENTS

For the purposes of this methodology, an information system is viewed as the sum of its users and their goals, and the system's services and their subordinate activities. This is illustrated in Figure 2. It is assumed that the system's analyst can identify and classify the system's users according to their service request characteristics and according to the performance constraints imposed upon the system (i.e., goals) when honoring their requests. It is also assumed that the analyst can identify those information system activities which are critical to system performance. Obviously, the complexity of the problem is increased substantially when a system supports diverse users or provides a wide spectrum of services. Whether or not the system is complex or simple, the criteria for identifying system activities should be based upon the sensitivity of the system's performance with regard to changes in their behavior.

Information system services are viewed as being a series of distinct yet interconnected activities which are invoked during the processing of a stream of user requests for the

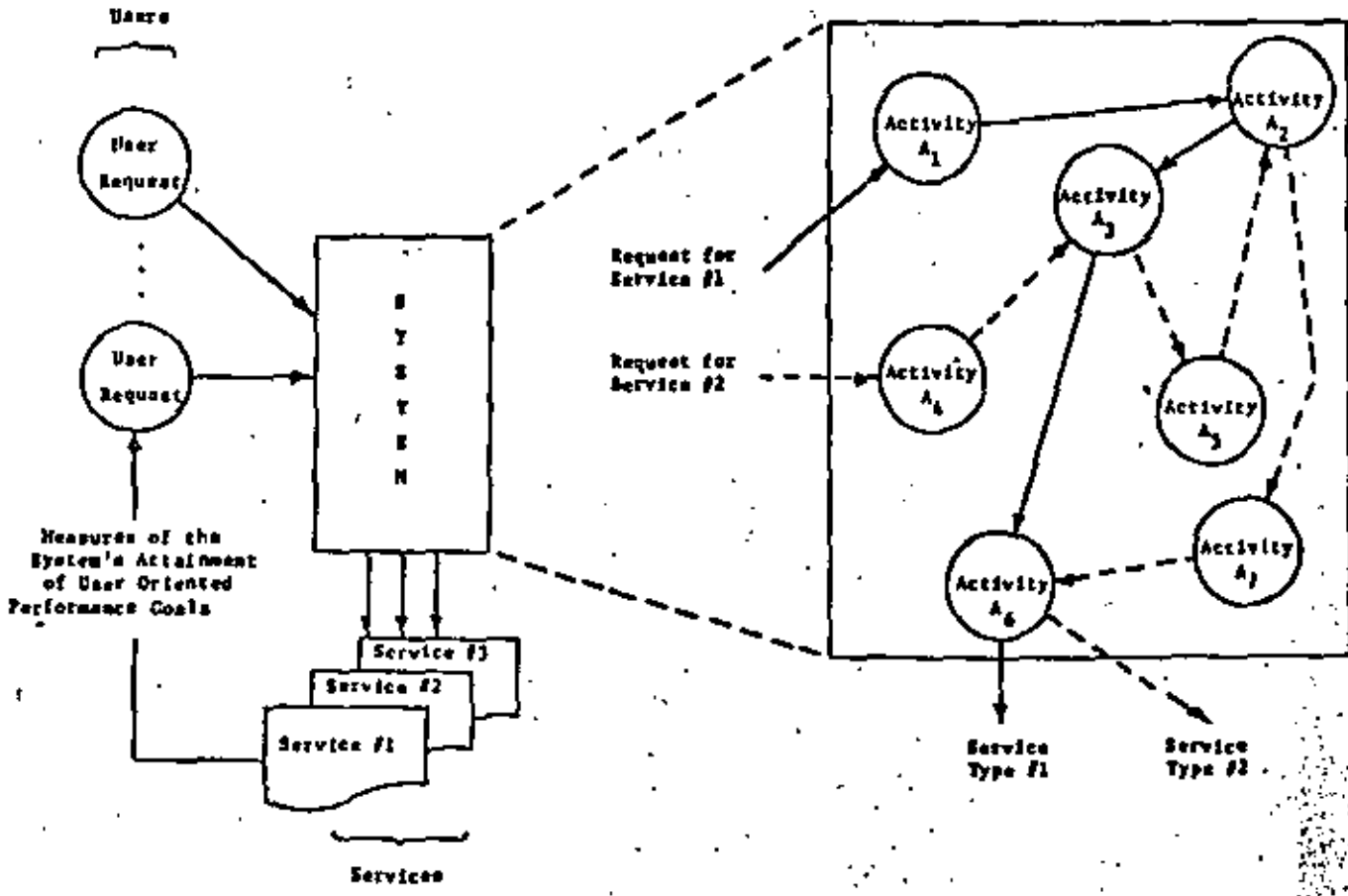


Figure 2—The methodology's view of an information system

service. Again, Figure 2 illustrates this view of a system. Most likely, system activities are aggregations of one or more traditional computer system functions that perform the following tasks:

1. request (job) scheduling,
2. task management,
3. resource allocation,
4. secondary storage I/O processing, and
5. application processing.

The choice of what constitutes an activity is part of the art of performance evaluation, however, a necessary condition for their selection is that they be measurable and that these measurements distinguish the service rates for separate classes of system services. It is also assumed that the role of performance measurement is to determine the current processing rate for the jth activity with respect to the ith service.

Figure 3 is a schematic of the functional composition of system activities. Each is viewed as an individual queueing system containing one or more priority queues and one or more identical servers. Additionally, the performance measure for the activity in processing a request type is the sum of both the queue performance and service functions of the activity. Throughout this paper, the variable $R_j(i)$ is employed to identify this performance of activity A_j with respect to Service S_i. It is assumed to be the average of performance for all the executions of A_j for S_i. Also associated with each activity A_j is the performance factor β_j which is interpreted as the scaling factor to be applied to the $R_j(i)$ to obtain the level of performance for the jth activity which minimizes the goal programming objective function. It should be noted that the problem of identifying a "good" level of performance for activities, i.e., determining the appropriate values of the $R_j(i)$'s is compounded by the multiple use of the activity by different and possibly conflicting services. Therefore, the modification of an activity's processing rate to achieve one goal may be counterproductive to the attainment of another goal. It is on this

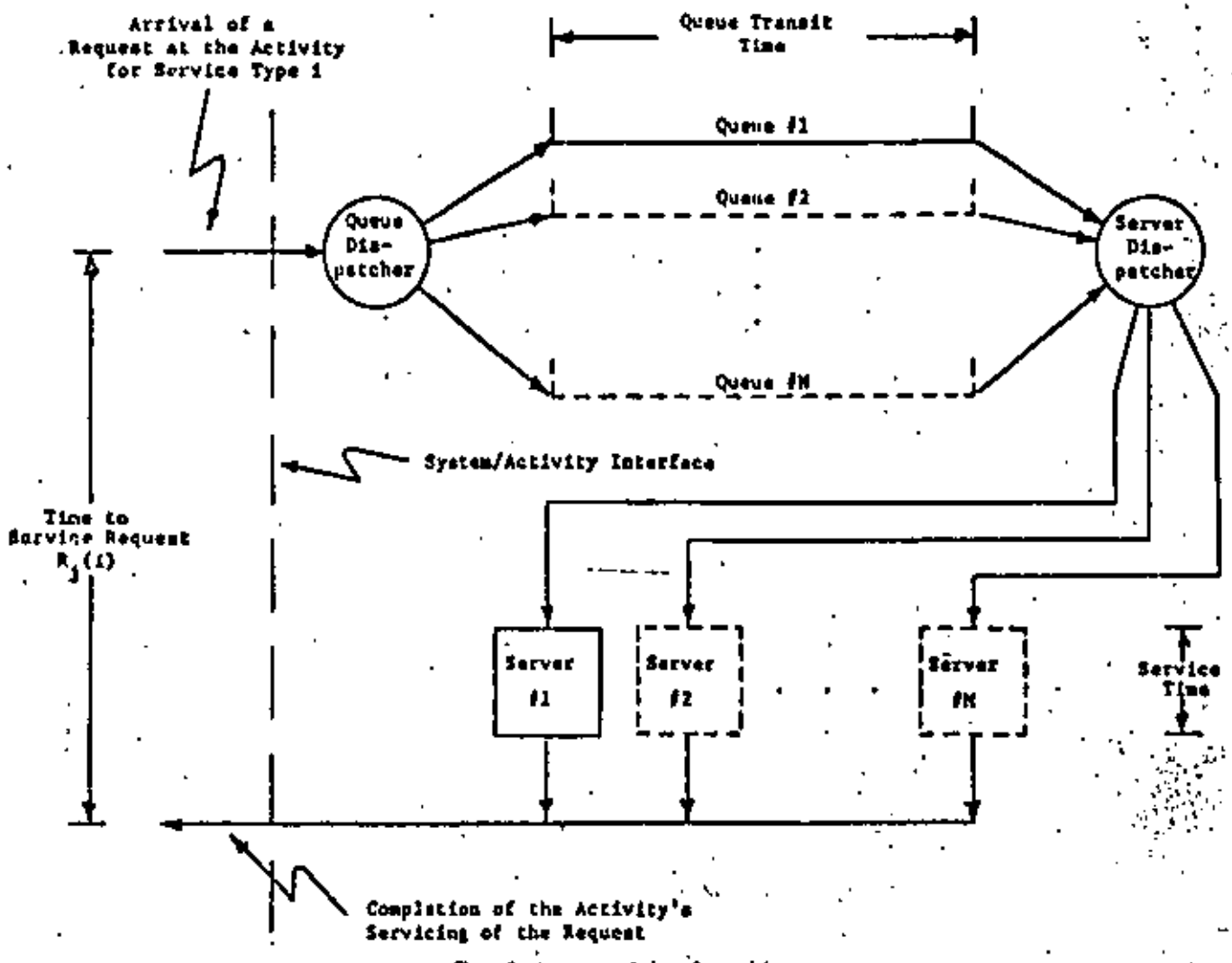


Figure 3—A conceptual view of an activity

possibility of multiple conflicting interactions and goals that this methodology is focused.

FORMULATION OF THE STAGE 2 EVALUATIVE PROCEDURE

Stage 2 is based on an evaluative procedure commonly called multiple goal programming (MGP). The procedure was first formulated by Charnes and Cooper[22] in 1961 to solve linear programming problems that had conflicting constraints. Ijiri[23] developed the details of the procedure within the framework of mathematical programming. This technique has been used to solve problems in the areas of strategic management planning such as accounting control,[24] advertising-media planning (Charnes and Cooper[25]), and resource allocation.[26] The employment of goal programming in conjunction with information system performance evaluation is a new use of the procedure.

There are three reasons for choosing multiple goal programming for use in this stage of the methodology. First, this approach can evaluate linear and ordinal multiple goal situations, both of which are inherent to information systems evaluation. For example, one user class may pay twice as much for its service, and, therefore, satisfaction of its goals may be worth twice as much as others; a linear relation. On the other hand, certain users, such as a critical patient monitoring application, may have incomparable importance relative to other classes; an ordinal relation. Second, multiple goal programming produces a solution that not only evaluates the total goal situation, but also evaluates each goal, individually. One of the purposes of this methodology is to determine the critical user classes and associated activities. Third, MGP derives the "best" design under the given goal constraints. Other design approaches such as weighting, sequential elimination, and spatial proximity,[27] are based on selecting the "best" design from a finite set of alternatives. The purpose of the overall methodology, however, is to *design* an appropriate system to satisfy the user and resource constraints. The standard formulation of a multiple goal programming problem is:

[A] Minimize $P \cdot D$
Subject to $A \cdot X + D = G$
where
- A = a matrix of technological coefficients which can be thought of as the rates at which the ith service uses the jth resource
- X = the array of resulting system resource allocation levels
- G = the array of service goals
- D = the array of discrepancies from these goals
- P = the array of penalties associated with the discrepancies in D

and where the objective function is to minimize the product of the discrepancies and their associated penalties. The solution to a multiple goal programming problem represents the best set of levels for the resource allocation vector X such that the objective function is minimized. The remainder of this section discusses the specific formulation for the Stage 2 component of the methodology.

Figure 4 illustrates the relationship between MGP, the information system activities, and its servicing of user requests. The servicing of a request type i is a sequence of activities, $A_1, A_2, \ldots, A_n$, each assumed to be measurable by $R_j(i)$. In general, the measure can be a function $M(R_j(i))$ of the service time, however, just $R_j(i)$ will be employed in the following discussion.

The performance of the information system for service type i is given by the relation

$$T_s(i) = \sum_{j=1}^{n} (R_j(i)) \tag{1}$$

where $T_s(i)$ is the average system response time to service requests of type i. Assuming that the performance goal for the service is $T_g(i)$, then the discrepancy between performance and goal is given by

$$D(i) = T_s(i) - T_g(i). \tag{2}$$

The objective of MGP is to determine new performance levels for each activity in such a manner that the weighted discrepancy, $P \cdot D$, is minimized (hopefully to zero). Letting β_j be a scaling factor to be applied to the jth activity, then the new performance level for the activity is $R_j(i)\beta_j$. Incorporating the β_j's into equation (1) results in the following expression for the discrepancies:

$$D(i) = T_s(i) - \sum_{j=1}^{n} (R_j(i) \cdot \beta_j). \tag{3}$$

Observe that both positive and negative discrepancies are possible, and, therefore, the formulation of the user goal evaluation as a MGP problem becomes:†

[B] Minimize $P_0^+ \cdot D^+ + P_0^- \cdot D^-$
s.t. $R \cdot \beta + D^+ - D^- = G$
where
- R is a matrix of service rates
- β is the array of scaling factors
- G^+ is the array of user goals
- D^+ and D^- are the arrays of, respectively, the positive and negative discrepancies from the user goals
- P^+ and P^- are the arrays of penalties associated with the corresponding positive and negative discrepancies.

The formulation serves two purposes: it evaluates goal achievement and produces the β_j's. By setting the values of the β's to reflect only the current configuration (i.e., $\beta_j = 1$),

† The complete derivation appears in a previous paper by the authors presented at the Annual Conference of the Computer Measurement Group, November, 1976.[28]

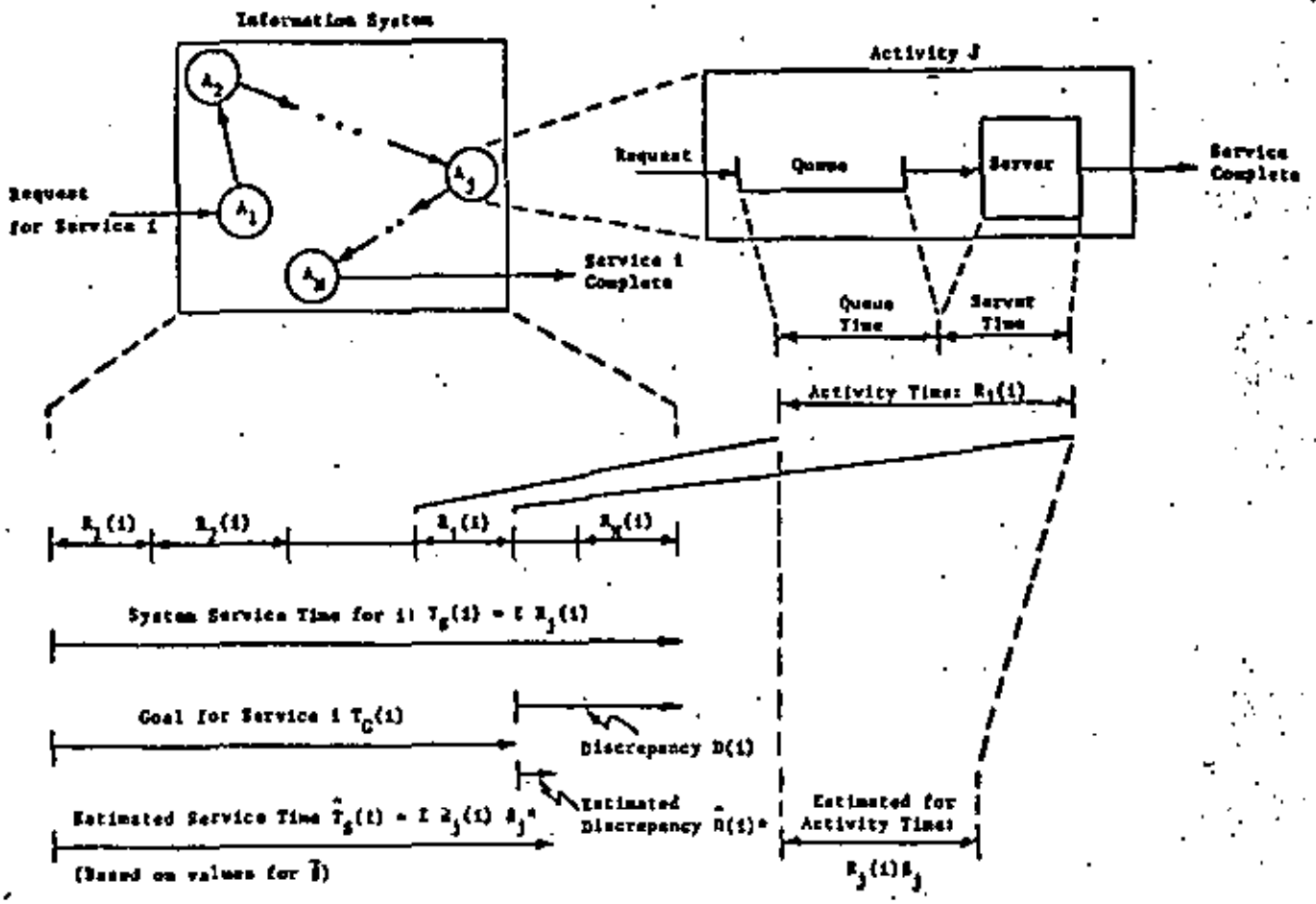


Figure 4—Relationship between goal programming and information systems characteristics

the evaluation of the system's attainment of the user goals is accomplished.

Experiments with formulation [B] produced valid; but impractical sets of β's. The MGP problem as stated allowed for the possibility of solutions where a β_j could equal 0, clearly an unacceptable situation. In order to inhibit this type of solution, limits were placed on the range of possible β_j values. This was accomplished with the following set of additional constraints:

$$\beta_j+\mu_j^- -\mu_j^+=L_j \quad (4)$$

$$\beta_j+\nu_j^- -\nu_j^+=H_j \quad (5)$$

where

$$0<L_j\leq 1$$

$$1\leq H_j$$

L_j is used to restrict the alternative possibilities for the case that $\beta_j<1$ while H_j is used for those cases that $\beta_j>1$. In general, the set of all positive discrepancies for L_j's, (μ_1^+, μ_2^+, . . . , μ_j^+)=M^+ and likewise for H_j(ν_1^+, ν_2^+, . . . , ν_j^+)=N^+. (M^- and N^- have similar definitions).

These constraints are reflected in the objective function in a manner different than previous constraints. Instead of minimizing both discrepancies, only one is minimized. In the case of L_j only μ_j^- is included, since, if μ_j^- is driven to zero, then $R_j(i)\beta_j-\mu_j^+=L_j$, implying that $R_j(i)\beta_j>L_j$, the desired condition. Similarly, for H_j only ν_j^+ is in the objective function because minimizing ν_j^+ results in $R_j(i)\beta_j<H_j$.

These added constraints also have a physical interpretation relative to the evaluation of the system. No activity can be eliminated from a system (i.e., $\beta_j=0$) since L_j must be greater than 0. In general, however, L_j represents the lower bound on the degree of reduction feasible for the current rate of usage for an activity. For example, $L_j=.25$ implies that the usage rate for activity j can be made, at most, four times faster, being reduced to 25% of its current rate. Similarly, H_j represents the upper bound on the degree to which an activity's rate can be increased (slowed down). It must be emphasized that these limits are only rough estimates, not exact values.

In order to reduce the number of alternatives one should minimize the number of modifications indicated per evalua-

tion iteration. Since modifications are characterized by the production of β_j's not equal to 1, a secondary objective of Stage 2 is to produce as few $\beta_j \neq 1$ solutions as possible. This is accomplished by including the constraint equation

$$\beta_j + \epsilon_j^- - \epsilon_j^+ = 1 \qquad (6)$$

while minimizing both ϵ_j^+ and ϵ_j^-

Constraints of this type provide a default value of 1 for the multiple goal programming procedures in the case where an activity is neither critically inefficient or excessive. (Note: $(\epsilon_1^+, \epsilon_2^+, \ldots, \epsilon_j^+) = E^+$.)

As a result of these added constraints, the actual formulation of the MGP problem used in Stage 2 is given in formulation [C] below:

$$\begin{aligned}
[C]\ \text{Minimize}\quad & P_D^+ \cdot D^+ + P_D^- \cdot D^- + P_M^- \cdot M^- + P_M^+ \cdot N^+ + P_E^+ \cdot E^+ + P_E^- \cdot E^- \\
\text{s.t.}\quad & R \cdot \beta + D^- - D^+ = G \\
& \beta + M^- - M^+ = L \\
& \beta + N^- - N^+ = H \\
& \beta + E^- - E^+ = I
\end{aligned}$$

where

- R is the matrix of service rates
- β is the array of scaling factors
- G, L, H, and I are the arrays of goals for the user criteria and the respective β constraints
- D^*, M^*, N^* and E^* are the arrays of positive and negative discrepancies from the respective goals
- P_D^*, P_M^*, P_N^*, and P_E^* are the arrays of penalties for the associated discrepancies.

The solution variables for the MGP problem are the β's. They identify those activities that must be altered in order to improve user based or system based performance. If the value for a $\beta_j = 1$ then the service characteristics of activity j were adequate to satisfy all the user's criteria. If a $\beta_j < 1$, this implies that the current service rate of activity j is insufficient to meet the system's needs. The new service rate for the activity should be $R_j(\cdot) = (\beta_j) * (R_j(\cdot))$. If a $\beta_j > 1$ then the current service rate of activity j is faster than necessary and there exists the possibility of excess capacity. The new unit rate should be $R_j(\cdot) = (\beta_j) * (R_j(\cdot))$.

Assuming that an activity follows the characterization in Figure 3, then the analyst has three avenues of action when a $\beta_j \neq 1$. First, he can analyze the queue dispatching discipline in order to increase queue throughput (or possibly replace it with a simpler one if $\beta_j > 1$). Second, he can alter the service rate characteristics of the servers, e.g., slower hardware devices. And third, he can increase (decrease) the degree of parallelism among servers, for example, by adding (removing) a second channel, controller, etc.

LIAISON WITH STAGE 1

The critical factors in the Stage 2 evaluation are the values for the $R_j(i)$'s needed by the MGP formulation. These values are calculated in Stage 1 and are the statistical measure produced vis-a-vis the simulation. The specific model to be evaluated is the result of the heuristic procedures constituting Stage 3. The liaison is based upon the assumption that an information system can be viewed as a collection of resource allocation and task management activities and user oriented services. This view is supported by the literature, e.g., Madnick[28] and Zurcher and Randall.[29] IPSS also views the modeling of an information system in a similar manner, and thus, facilitates the development of the formal liaison with the MGP user evaluation.

The view of the system taken in Stage 2 (as illustrated in Figure 2) has an analog in IPSS. Its basic modeling concept is that of a service as shown in Figure 5. The service is classified in IPSS as a procedural facility and is capable of representing any information system activity including request (job) scheduling, task management, resource allocation, secondary storage I/O processing and application software. Since services are allocatable facilities in IPSS they have associated with them both queueing and utilization statistics. Furthermore, service behavior can be predicated on the quantity and characteristics of other IPSS hardware and software facilities. Therefore, the statistics associated with service facilities have the appropriate structure to service as the R_j's needed in Stage 2.

To complete the formal liaison between the two stages a second feature is employed. This is the Task facility. Through its use, the service facility statistics can be automatically segregated into service statistics by user. In this manner, the statistic $R_j(i)$, required by Stage 2, is collected. Thus, the Stage 2 users (indexed by i) and the activities (indexed by j) are, respectively, an IPSS model's Task and Service facilities. An $R_j(i)$ is the sum of queueing and utilization statistics gathered for Service i when executing Task j.

Stage 1 must also be adaptive to model changes dictated via Stage 3. Again, the IPSS model synthesis philosophy and language constructs permit the desired adaptiveness. This is possible for reasons too detailed to discuss in this paper. A complete description of IPSS is available in the document titled "The Information Processing System Simulator (IPSS): Language Syntax and Semantics."[30] Briefly, however, possible modifications to an existing and executing model without requiring complete reformulation include

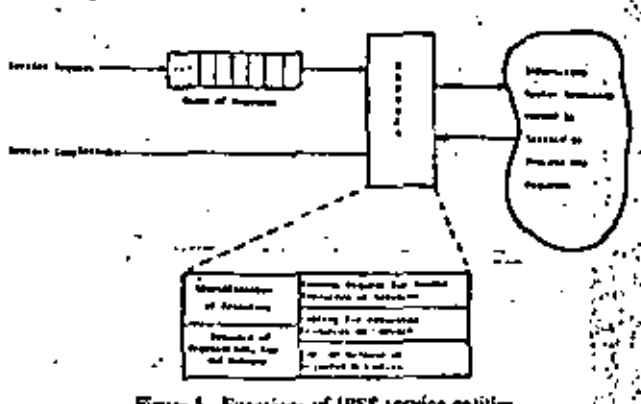


Figure 5—Functions of IPSS service entities

changes to:

1. timing and space characteristics associated with secondary storage hardware and storage media,
2. the secondary storage I/O configuration,
3. user usage patterns and service requirements,
4. file organization methods and space management policies,
5. the queueing disciplines associated with job scheduling, resource allocation and task management, and
6. memory management policies.

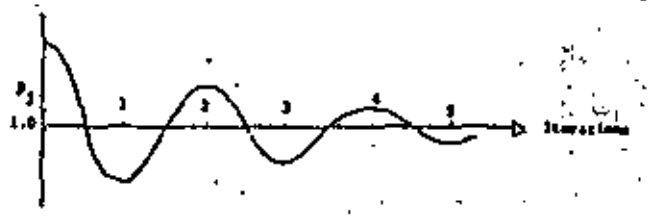
IPSS supplies the Stage 1 processing with a capability of being self-adaptive with respect to Stage 3 outputs. Currently, the methodology employs modeler assistance in Stage 3. Future research will be directed at providing more sophisticated heuristics for Stage 3 in order to provide a truly self-contained iterative methodology for the multiple criteria evaluation of information systems.

STAGE 3 ANALYSIS

The functions of Stage 3 of the methodology are to determine whether the current configuration is satisfactory and to formulate a new model in light of the data provided from Stage 1 and Stage 2. Figure 6 shows the information flow to Stage 3. New models reflect the performance goals of both the user and the system. Heuristics using Sutherland's* definition of a heuristic are employed in Stage 3.

The problems encountered are complex and unstructured. Determining if the current design is acceptable requires a mixture of objective and subjective reasoning. It would be a rare situation if all the user goals and system constraints were satisfied simultaneously. Generally, an extremely wide spectrum of acceptable performance levels and alternative designs are possible, at each iteration, to satisfy both user and system criteria. Trade-offs will dominate the decision processes. Many factors effecting suitable designs may not be included in the formulations and procedures of the first two stages. For example, there may be external political, organizational or economic considerations that are not directly related to the performance of the system, but may be a major factor in the final decision. The methodology does assume, however, that the heuristic procedures have access to this external criteria.

When it has been determined that another iteration is desirable, it is assumed that the heuristics will examine current and past designs. Whatever the heuristic employed, ideally its objective is to produce a sequence of models whose β_i characteristics (for all β's) behave as follows.



The emphasis of the current research is to provide insight into the decision process for improving the performance of information systems. Stage 3 is this decision process. It is aided by input from four sources within the methodology. These sources are: (a) the Stage 2 outputs, specifically the β_i's identified to improve user and system goal performance, (b) system behavior statistics from Stage 1, (c) the current model, and (d) historical data from prior iterations. It should be emphasized that at this juncture in the development of the methodology no formal heuristic procedures have been implemented. It is one of the purposes of this research, however, to investigate the appropriateness and success of various heuristic decision rules. Rules of thumb such as those proposed by Buzen[21] are possible avenues to be investigated.

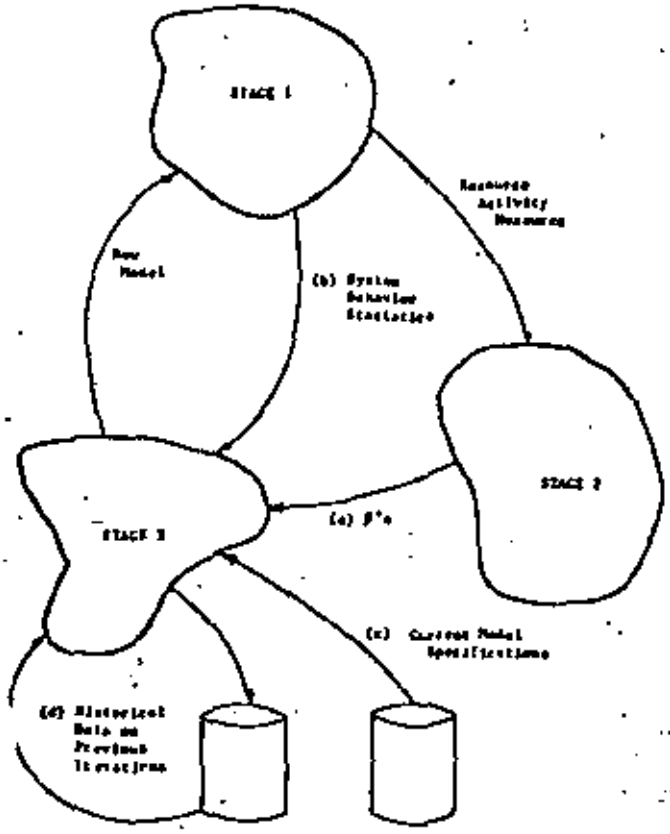


Figure 6—Information flow to stage 3

AN EXAMPLE

In order to validate the procedures developed in this methodology and the liaison between Stage 1 and Stage 2, a test case was developed. This example was modeled and executed in IPSS to satisfy the Stage 1 requirements.

* A heuristic is "a disciplined trial-and-error process . . . in exercise in successive improvement, where we may learn from both success and failure and where the criteria for success and failure may vary with what we have previously learned" (p. 183, Reference [29]).

TABLE I—User Goal Evaluation

Goal	Over-Achievement	Under-Achievement
$USER_1$	26.6	0.0
$USER_2$	0.0	51.3
$USER_3$	0.0	258.5

Several iterations were applied, demonstrating the evaluative capabilities of the methodology. The following is a description of the problem, the corresponding model and the results of the first two iterations.

The example is a model of an on-line document retrieval system. There are three files associated with the system; an author/title index (A/T), a system document identification file (ID), and the document file itself (DOC). They are structurally related such that an entry in the A/T file points to one or more entries in the ID file and each ID file entry in turn is associated with only one DOC entry.

The model is designed so that a unique activity is associated with the accessing of each file; Activity 1 with the A/T file, Activity 2 with the ID file and Activity 3 with the DOC file. Each activity performs similar functions: obtaining and releasing devices, reading records, performing I/O techniques, but to different files.

It is assumed that the system supports three user classes, each with a different demand on the retrieval system and each characterized by a different combination of activities. The purpose of the first user class, $USER_1$, is to retrieve a document for a particular author, thus utilizing all three activities. Those in the second user class, $USER_2$, want to determine the existence/non-existence of an entry in the system for a given author, and therefore, need to use only Activity 1. The final user class, $USER_3$, already has the address of the ID entry and wants to retrieve the associated DOC entry requiring only Activity 2 and Activity 3.

In order to complete the formulation of the performance evaluation problem for this methodology, assumptions concerning the performance of the system were made. A summary of these assumptions for the user goals and β constraints and the penalties associated with the corresponding discrepancies in accordance to the requirements of formulation [C] is shown below.

$$G=(\ 150.0,\ 100.0,\ 400.0)$$
$$L=(\ .2,\ .2,\ .2)$$
$$H=(\ 5.0,\ 5.0,\ 5.0)$$
$$I=(\ 1.0,\ 1.0,\ 1.0)$$
$$P_G^+=(1000.0,\ 10.0,\ 1000.0)$$
$$P_G^-=(\ 1.0,\ 0.0,\ 0.0)$$
$$P_H^+=P_K^-=P_L^+=P_L^-=(\ 1.0,\ 1.0,\ 1.0)$$

The model constructed in IPSS assumed a simple configuration of one processor and one bank of IBM 2314 type direct access devices. Under a given loading (which is not a controllable variable in this methodology) the resulting performance statistics, the values of matrix R, are shown below.

$Q_1(1)=0.0$ $Q_2(1)=0.0$ $Q_3(1)=11.4$
$S_1(1)=37.2$ $S_2(1)=37.1$ $S_3(1)=90.9$
$R_1(1)=37.2$ $R_2(1)=37.1$ $R_3(1)=102.3$

$Q_1(2)=12.2$ $Q_2(3)=0.0$ $Q_3(3)=8.9$
$S_1(2)=36.3$ $S_2(3)=39.1$ $S_3(3)=93.5$
$R_1(2)=48.5$ $R_2(3)=39.1$ $R_3(3)=102.4$

This performance information, coupled with the goal assumptions, was input to Stage 2. The evaluation of the current configuration's performance with respect to the set of user's goals is shown in Table I. It indicates that the goals of user classes 2 and 3 were satisfied with a good margin of slack, (which is not penalized in this example) while the goal of the first user class was not satisfied (over-achievement implying non-satisfaction).

In the second phase of Stage 2, the β's are allowed to be manipulated until they satisfy the user goal constraints and best suffice the system guideline constraints. The result is the identification of those activities whose performance can be, and need to be, improved with respect to one or both of the criteria. The values for the β's as calculated were

$$\beta_1=1.0 \qquad \beta_2=1.0 \qquad \beta_3=.74$$

These are interpreted as indicating that both Activity 1 and Activity 2 were adequate to meet the user demands put to them. Activity 3, however, was found to be insufficient to satisfy the requirements of user classes 1 and 3. The modification indicated is to reduce the present rate of usage for Activity 3 by at least 1/4 in order to satisfy the user goals, in particular, the first user class goal.

The determination of whether to cease the design loop by accepting this performance or to continue by modifying the existing model is made in Stage 3. Given the stated goal/penalty structure, it was assumed that the over-achievement of $USER_1$ goal was at an unacceptable level and the design process must continue if possible. By examining the queueing and service time statistics for the first iteration, one can eliminate some of the modification possibilities. The result of Stage 3 analysis was a decision to replace the IBM 2314 type device with a faster one, i.e., an IBM 3330 type device.

The original model of this example system was dynamically altered to reflect this modification. Under the same loading as before, the following performance statistics were accumulated.

$Q_1(1)=0.0$ $Q_2(1)=0.4$ $Q_3(1)=3.8$
$S_1(1)=11.9$ $S_2(1)=11.3$ $S_3(1)=27.6$
$R_1(1)=11.9$ $R_2(1)=11.7$ $R_3(1)=31.4$

$Q_1(2)=3.8$ $Q_2(3)=0.0$ $Q_3(3)=0.0$
$S_1(2)=11.1$ $S_2(3)=12.0$ $S_3(3)=29.1$
$R_1(2)=14.9$ $R_2(3)=12.0$ $R_3(3)=29.1$

Stage 2 analysis showed that now all three user class goals were satisfied (i.e., not over-achieved). The calculation of the β's, however, indicated that while Activities 1 and 2 were still adequate ($\beta_1=\beta_2=1.0$), Activity 3 now had the possibility of excess capacity ($\beta_3=4.0$). Although a slower and probably less expensive device for Activity 3 would be more appropriate, we had found in the first iteration that such a device was not able to satisfy all the user goals. Therefore, in future iterations, Stage 3 procedures had to examine more subtle methods of improving performance.

CONCLUSION

Modern information systems do not exist as entities unto themselves, but must interact with their environment, i.e., their users. The loading and mix of the users effect the performance of the system resources and likewise, the service characteristics of the system resources effect the satisfaction of user goals. In order to design such systems, one must satisfy a large set of users demanding a conflicting set of performance goals while operating within efficiency and minimum cost constraints. Thus, performance evaluation of information systems is a multiple criteria problem. Concurrently satisfying both of these sets of criteria is the goal of this methodology. Current available techniques, however, only address one side of the problem, either the user or the system. The methodology described in this paper establishes a formal liaison between the evaluation of user goals as a function of system behavior and the analysis of system resource performance as a function of user demand, thereby, facilitating multi-criteria evaluation.

The methodology is iterative, comprising three separate but integrated stages. The first stage models and evaluates system behavior. The particular technique employed in the first stage is IPSS and it is able to collect the necessary statistic, $R_i(i)$. The second stage evaluates the user based criteria and provides evaluative insight into performance improvement. Solution of the MGP formulation in [C] produces a set of β's, the variables of Stage 2 which indicate inefficiencies and/or excesses in the current model. And finally, the third stage heuristically determines the current model's acceptability and need for modifications.

The evaluative procedures developed for this methodology have been shown to be valid in practice. Furthermore, this methodology provides an excellent basis for continued research into areas such as:

a. investigation into the causal relationships between user demand and system activity,
b. sensitivity analysis of these relationships,
c. investigation into suitable heuristics for Stage 3, either testing existing heuristics or development of new ones, and
d. development of heuristic/modification rules to close the design loop into an automatic self-modifying process.

REFERENCES

1. Boehm, B., and T. E. Bell, "Issues in Computer Performance Evaluation: Some Concensus, Some Divergence," *PER*, Vol. 4, No. 3, 1975, pp. 4-39.
2. Gaver, D. P., and G. Hunsfeld, "Multitype Multiprogramming: Probability Models and Numerical Procedures," *Proc. of CPMME*, 1976, pp. 38-43.
3. Buzen, J. P., "Computer Algorithms for Closed Queuing Networks with Exponential Servers," *CACM*, Vol. 16, No. 9, 1973, pp. 527-531.
4. Neilson, J. E., "An Analytic Performance Model of a Multiprogrammed Batch Time-Shared Computer," *Proc. of CPMME*, 1976, pp. 59-70.
5. Cooper, C. R., "The Simulation and Evaluation of Information Retrieval Systems," Report 352-R-17, April 1965.
6. Roehrkasse, R. C., and D. Smith, "Simulation of Operating Systems," Tech. Report GITIS-70-11, 1970, Georgia Inst. of Tech.
7. Abate, J., H. Dubner and S. B. Weinberg, "Queuing Analysis of the IBM 2314 Disk Storage Facility," *JACM*, Vol. 15, No. 4, 1968, pp. 577-589.
8. Nabouni, E., "Direct Access Device Simulation," *IBM Systems Journal*, Vol. 13, No. 1, 1973, pp. 19-31.
9. Sherman, S. W., and R. C. Brie, "I/O Buffer Performance in a Virtual Memory System," *Symposium on the Simulation of Computer Systems*, 1976, pp. 24-35.
10. Heffernan, H. R., and H. J. Smith, "Throughput Analysis of Some Idealized Input, Output and Computer Overlap Configurations," *Computing Surveys*, Vol. 2, No. 2, 1970, pp. 111-118.
11. Kleinrock, L., and R. R. Muntz, "Processor-Sharing Queueing Models of Mixed Scheduling Disciplines for Time-Shared Systems," *JACM*, Vol. 19, No. 3, 1972, pp. 464-482.
12. Agrawala, A. K., and R. L. Larson, "Experience with the Central Server Model on a Lightly Loaded System," *Symposium on the Simulation of Computer Systems IV*, 1976, pp. 183-189.
13. Lewis, P. A. W., and G. C. Shedler, "A Cyclic-Queue Model of System Overhead in Multiprogrammed Computer Systems," *JACM*, Vol. 18, No. 2, 1971, pp. 199-220.
14. Gonzalez, G., "Using Covariance Analysis as an Aid to Interpret the Results of a Performance Measurement," *Proc. of CPMME*, 1976, pp. 179-186.
15. Cantrell, H. N., and A. L. Ellison, "Multiprogramming System Performance Measurement and Analysis," *AFIPS Conf. Proc.*, Vol. 32, 1968, pp. 213-221.
16. Reeves, T. E., and U. W. Pooch, "A Multiple Subsystem Simulation of Processor Scheduling," *Symposium on the Simulation of Computer Systems III*, 1975, pp. 129-135.
17. Norland, K. E., and W. C. Balgrph, "A Simulation Model of GECOS III," *Proc. of ACM*, 1971, pp. 596-612.
18. Lum, V. Y., Ling, H., and Senko, M. E., "Analysis of a Complex Data Management Access Method by Simulation Modeling," *FJCC*, 1970, pp. 211-222.
19. Lindsay, D. S., "A Hardware Monitor Study of a CDC KRONOS System," Proc. of CPMME, 1976, pp. 179-186.
20. Hall, W. A., "A Simulation Model to Aid in the Design and Tuning of Hierarchical Databases," *Winter Simulation Conference*, 1974, pp. 277-284.
21. Buzen, J. P., "Fundamental Laws of Computer Performance," in *Proc. of Int'l. Symp. on Computer Performance Modeling, Measurement and Evaluation* (CPMME), 1976, pp. 200-210.
22. Charnes, A., and W. W. Cooper, *Management Models and Industrial Applications of Linear Programming*, New York, John Wiley & Sons, Inc., 1961.
23. Ijiri, Y., *Management Goals and Accounting for Control*, Chicago, Rand McNally, 1965.
24. Charnes, A., et al., "A Goal Programming Model for Media Planning," *Management Science*, Vol. 14, No. 8, April 1968, pp. 423-430.
25. Lee, S. M., *Goal Programming for Decision Analysis*, Philadelphia, Auerbach, 1972.
26. MacCrimmon, K. R., "An Overview of Multiple Objective Decision

Making," *Multiple Criteria Decision Making*, eds., J. L. Cochrane and M. Zeleny, Columbia, S. C., 1973, pp. 18-44.
27. Chandler, J. S., and T. G. DeLutis, "A Methodology for the Performance Evaluation of Information Systems Under Multiple Criteria," *Proc. of Computer Measurement Group*, 1976, pp. 221-230.
28. Madnick, S., and J. Donovan, *Operating Systems*, N. Y., McGraw-Hill, 1974.
29. Zurcher, F. W., and B. Randell, "Iterative Multi-Level Modelling: A Methodology for Computer System Design," *IFIP 68*, pp. 867-871.
30. DeLutis, T. G., "The Information Processing System Simulator (IPSS): Language Syntax and Semantics," unpublished research report (Grant No. G-36622).
31. Sutherland, J. W., *Systems: Analysis, Administration, and Architecture*, Van Nostrand Reinhold, New York, 1975.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

APPLICATIONS OF SPARCOM DATA BASE CONCEPTS TO A CRIME COMBATING ENVIRONMENT

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

Applications of SPARCOM data base concepts to a crime combating environment*

by RON ASHANY
Thomas J. Watson Research Center
Yorktown Heights, New York

ABSTRACT

In this paper only certain aspects of a powerful Data Base System called SPARCOM, as reflected by some applications in a crime-combating environment are presented. SPARCOM stands for Sparse Associative Relational Connection Matrix. It is a method that was developed for the analysis, interpretation, organization, classification, update, and structure of stored data as well as for the search and retrieval of information from large data base (LDB) systems. The unique approach of this system is the conversion of data into large sparse binary matrices that enables one to apply sophisticated sparse matrix techniques to perform data base operations. The operations are performed on the matrices as though the entire matrix were present, but great amounts of storage space are saved, and execution time is significantly reduced by the storage and manipulation of the nonzero values only. Additional reduction in storage requirements and in execution time is achieved by SPARCOM's intrinsic normalization process that alleviates the grave problem of data redundancy, caused by multi-value attributes.

INTRODUCTION

At the current state-of-the-art, Data Base systems are still largely the result of ingenious casual and often belated attention to human factors, users' needs, system configurations and programming problems. The methods used in the development and design of such systems are essentially trial-and-error. No evidence is to be found in the technical literature, describing existing systems, to indicate that the development was based on a sound scientific foundation or on well established engineering disciplines. Generally speaking, no unifying concept, general theory, or common systematic approach is as yet evident in work in the field of data base technology. One of the major contributing factors to the haphazard approach is the lack of distinction, during the development phases, between physical and logical considerations.

Before contemplating the development and design of any system, an appropriate conceptual framework must be established. To understand and communicate the concepts of such a framework certain definitions must be postulated. Some of the definitions are precise and have been established on rigorous mathematical foundations, others are of a generic nature and still others have been established to provide a common base of reference. In the new field of data base technology, there is no evidence of an existing commonly agreed nomenclature. Many terms, introduced in this paper are new, others were adopted from different sources.

BASIC CONCEPTS AND DEFINITIONS

The digital computer cannot deal with physical objects, optical images, events or abstract concepts directly. A data model of the physical world or of the abstract concepts which can be easily manipulated must be constructed first. The *Data Model* (DM) is the information content of the data base as it is viewed by the users. For example, a Regular Data Organization (RDO) in an array form as illustrated by the criminal file in Figure 1, is an accepted form of representing data; therefore, it is a data model.

A collection of processable data from which an enterprise can derive information is called a *Data Base*. A police department, a hospital, a court of justice, a university, a bank or any large-scale administrative, scientific, technical, commercial or other type of operation is called an *Enterprise*. An item of interest to an enterprise about which data are recorded or computed is called an *Entity*.

In any given enterprise, there may exist different types of entities, a criminal, a detective, a bullet's signature, a car, a firearm, a crime, a portrait, a report, etc. Entities are identified via their attributes. An *Attribute* is a specific class of information about an entity; each entity has at least one attribute. Because of their wide and frequent usage, a number of attributes have been given names such as: Age,

* This work represents applications of some concepts investigated as part of the research, reported in the dissertation by R. Ashany, *SPARCOM: A Sparse Matrix Associative Relational Approach to Dynamic Data Structuring and Data Retrieval*, submitted to the Faculty of the Polytechnic Institute of New York in partial fulfillment of the requirements for the degree of Doctor of Philosophy (Electrical Engineering) June 1976.

ID	NAME	CRIME	ARM	LOCATION*	SEX	EYES	LANGUAGE
1	John	Murder	A	Detroit	M	Blue	English
2	Claude	Rape Arm. Rob.	B	N.Y.C.	M	Green	English French German
3	Robert	Hijack	D	Houston	M	Hazel	English German
4	Nancy	Homicide	A	Chicago	F	Blue	English German
5	Ingmar	Rape. Murder	D	Detroit	M	Blue	English Russian Swedish
6	Roberto	Murder	A	Chicago	M	Black	English Italian Spanish
7	Marcel	Kidnap. Hijack	C	N.Y.C.	M	Green	French German Spanish
8	Johanna	Drug Push Arm. Rob.	E	L.A.	F	Green	English French Spanish
9	Jurgen	Kidnap. Homicide	B	Detroit	M	Blue	English German Russian
10	Tom	Drug Push Arm. Rob.	C	L.A.	M	Hazel	English French Swedish

*(Crime Location)

Figure 1—Sample of regular data organization from a criminal file

Salary, Color, Sex, Weight, etc. There is a clear distinction between the *Attribute Name* and *Attribute Value*. An entity named *car* has an attribute named *color*, the value of the attribute is *blue*. The ordered pair Attribute-Value is called a *Property*. A collection of entities described by similar attributes is called an *Entity-Set*. If an attribute assumes a unique value for each entity in an entity set, it is called an *Identifying-Attribute*. An entity may have several identifying attributes; e.g., Full Name, Social Security Number, Employee Number, Prisoner Number, etc. The values of identifying attributes are called *Identifiers*. Attributes whose values can be calculated are called *Virtual Attributes*, such as Age; subtracting from the current year the year of birth the age value is determined.

In the regular data organization of the criminal file, illustrated in Figure 1, each row in the array belongs to a specific entity, $E_1, E_2, \ldots, E_m$. Each column belongs to a specific attribute; $A_1, A_2, \ldots, A_n$. Each element of the array is associated with a specific entity E_i and a specific attribute A_j and represents a specific value V_{ij} selected from

a finite set of values pertaining to attribute A_j. The set of values pertaining to attribute A_j is called the Domain D_j. The domain of attribute sex contains only two elements (M,F); the domain of attribute eyes (meaning eye color) contains five elements (black, blue, brown, green, hazel); the domain of attribute language is much larger; it contains all languages spoken in the world. A distinction should be made between single-value attributes and multi-value attributes. An entity can possess only one element from the domain of a *single-value* attribute, but it can possess one or more elements from the domain of a *multi-value* attribute. Sex and eyes are single-value attributes; language is a multi-value attribute. The multi-value attributes cause serious problems in some existing approaches for searching and retrieving data[1-3] but, they cause no problem whatsoever in SPARCOM.

The ordered triple (E_i,A_j,V_{ij}) represents an *elementary data item* which is the minimum amount of data required for meaningful information. The ordered pair (A_j,V_{ij}) represents the property P_{ij}. Consequently, the ordered pair (E_i,P_{ij}) represents an elementary data item. Let's illustrate the meaning of the minimum amount of data required for meaningful information: the entity John by itself does not represent meaningful information, the property Eyes Blue does not represent by itself meaningful information, but the triple John Eyes Blue is meaningful and precise. It is an elementary data item. The elementary data item (E_i,A_j,V_{ij}) is also known as the *Atom* of information because it cannot be further decomposed without losing its message.

MULTI-DIMENSIONAL ATTRIBUTE-SPACE MODEL

An entity can be represented as a point in a multi-dimensional Euclidean space by a vector. The six-tuple E_i(John,23,5.10,180,M,Blue) is a vector representing a point in a six-dimensional space and it can be interpreted as a person possessing the properties Name-John, Age-23, Height-5'10", Weight-180, Sex-M, Eyes-Color-Blue.

The notions of point, line, and plane from the familiar three-dimensional space may be generalized to higher dimensional spaces, and extended geometric interpretations of algebraic relationships may be given. Starting with a three-dimensional space, an entity set consisting of "m" entities can be envisioned as a set of points with corresponding coordinates described by a set of ordered triples $E_i(V_{i1},V_{i2},V_{i3})$ where $i=1,2,\ldots,m$. The values V_{i1},V_{i2},V_{i3} are selected from the respective domains of attributes A_1,A_2,A_3. For example, the attributes Age, Height, Weight can represent the respective orthogonal axes A, H, W. The first octant of the space is bounded by the three orthogonal planes: Age-Height, Age-Weight; and Height-Weight or AH, AW, HW, respectively. The triples

$$E_0(0,0,0);\ E_x(1,0,0);\ E_y(0,1,0);\ E_z(0,0,1)$$

represent the *Origin* and *Unit* points, respectively. To represent the aforementioned properties of the first four entities from a specific set, the following four triples are needed:

$$E_1(23,5.10,180);\ E_2(35,5.11,170);$$
$$E_3(27,6.02,210);\ E_4(19,5.04,130)$$

Each tuple represents a vector from the origin to a specific point in space. Each point can be projected on any of the three orthogonal planes. The 3-tuples

$$E_1(23,5.10,0);\ E_1(23,0,180);\ E_1(0,5.10,180)$$

represent the *projection* of the first point on the AH, AW, and HW planes, respectively. They also represent the *projection* of any points that may be found along the respective perpendicular lines through the points of projection. Clearly, any entity with properties Age 23 and Height 5'10" regardless of its Weight will be projected on the same point on the AH plane, the discussion is similar for the other planes. Much in the same manner the tuples

$$E_1(23,0,0);\ E_1(0,5.10,0);\ E_1(0,0,180)$$

represent the projection of the first point on the A, H, and W axes, respectively. They also represent the projection of any points that may be found on the respective planes that are perpendicular to the respective axes and intersect them at the points of projection. Actually, the triple $E_1(0,0,180)$ represents a plane parallel to the AH plane (perpendicular to the W axis) and intersecting the W axis at 180. The triple $E_1(23,0,180)$ represents a line parallel to the H axis (perpendicular to the AW plane) and intersecting the AW plane at the point (23,180).

QUERIES AND SEARCHING HYPERPLANES

Following the same line of reasoning, it may be said that in a three-dimensional space, a 3-tuple $Q(V_1,V_2,V_3)$ can be interpreted as a *Query* (an interrogation, a question) addressed to the data model. The request of the query is to search all entities represented by the points that coincide with the point specified by the coordinates of the query Vector Q. This is defined as the *Searching Point Query*. Each of the 3-tuples $Q(0,V_2,V_3)$; $Q(V_1,0,V_3)$; $Q(V_1,V_2,0)$ represents a query vector that requests the search of all entities represented by the points that their *projections* have the coordinates (V_2,V_3); (V_1,V_3); (V_1,V_2) on the orthogonal planes.

Obviously, in the multi-dimensional Euclidean space, the entities and queries are represented by *ordered N-tuples*. The five tuple $Q(V_1,V_2,V_3,V_4,V_5)$ represents a searching point query in a five dimensional attribute space. The tuples $Q(V_1,V_2,0,V_4,V_5)$ and $Q(V_1,0,V_3,0,V_5)$ represent searching lines and searching plane queries, respectively, while the tuples $Q(V_1,0,V_3,0,0)$ and $Q(0,V_2,0,0,0)$ represent 3-flat and 4-flat *Searching Hyperplane* queries respectively.

From this discussion, it should be obvious that the search is based on the content of the entities, rather than on their addresses, and that the entities that qualify are determined by the association that exists between the properties contained by the entities and those specified by the queries.

The query vectors must have the same dimensionality as the entity vectors searched by them. A collection of ℓ query vectors is called a *Query Set* and is represented by an $\ell \times n$ *Query Matrix*. *Query Complexity Degree* (QCD) is defined as the number of non-zero elements specified in the query vector. The QCD should not be confused with the *Request Complexity Degree* (RCD) which is determined by the type and number of properties, operators and operations required to get a complete response. The RCD will be defined in another section when the meaning of type of operators and type of operations will be better understood. To illustrate the meaning of QCD: the entity set defined by the attributes, Age; Height; Weight is interrogated by the following three requests:

Q_1(21,0,0)—Identify all criminals of age 21.
Q_2(35,5.11,0)—Identify all criminals of age 35 *AND* Height 5.11.
Q_3(27,6.02,210)—Identify all criminals of age 27 *AND* Height 6.02 *AND* Weight 210.
Q_1 is a *simple query* QCD=1 and it requires the search of only one domain.
Q_2 is a complex query QCD=2 and it requires the search of two domains, retrieving two subsets and retrieving the subset that results from the *intersection* of the previously retrieved two subsets.
Q_3 is a complex query QCD=3, the same as for Q_2, but with three domains.

If the *request* is to identify all criminals of age 21, *and* retrieve the names of all criminals who live in the same cities as those of the identified criminals, *and* speak Spanish, the request indeed starts with a simple query over one domain, but, imbedded in the request are complex queries that require the search of additional entity sets over different domains; obviously, the RCD should reflect the necessary interactions.

Another type of query known by the name of *Range Query* specifies a range of values over a domain rather than one specific value. For example, the request to identify all criminals between the age of 21 and 24, when the age is a discrete type attribute with a domain of positive integer numbers in increments of one year, the query has an QCD=4, because in this DM, the request has to be formulated as "identify all criminals of age 21 *OR* 22 *OR* 23 *OR* 24". In this data model, if there is a number r of *OR*s in the request, what appears to be one query has actually to be split into $q=(r+1)$ queries. The four 3-tuples representing the above mentioned request are

$$Q_1(21,0,0);\ Q_2(22,0,0);\ Q_3(23,0,0);\ Q_4(24,0,0)$$

and four subsets representing the points found on the four parallel planes will be retrieved. The subset obtained by the *union* of the retrieved subsets will represent the requested answer.

A similar situation will occur when the request deals with a *multi-value* attribute. For example, assuming that the entity set belongs to a 4-D (four-dimensional) attribute-space, and the request is to identify all criminals who speak English *AND* French, the query will have to be split into two 4-tuples

$$Q_1(0,0,0,\text{English}) \text{ and } Q_2(0,0,0,\text{French})$$

but the subset representing the result will be obtained by the *intersection* of the two subsets.

It is to be emphasized that the *AND* operators as opposed to the *OR* operators do not require the splitting of the query except when a multi-value attribute is involved. The request to identify all criminals who are males *AND* have blue eyes *AND* speak English *AND* German addressed to a 3-D entity set will be satisfied by splitting the query into two 3-tuples: i.e., Q_1(M,Blue,English) and Q_2(M,Blue,German). The split is caused by the *AND* of the multi-value attribute and the answer is obtained by the intersection of the two respective subsets.

SEARCH DIFFICULTIES IN THE RDO MODEL

In the RDO model where numeric and alphanumeric values appear in the same array it is very difficult to perform a search simply by comparing the corresponding elements of the entity vectors and query vectors. The search process can be greatly facilitated by encoding all alphanumeric values into pure numeric values. For example, the values of the crime attribute can be encoded so that Murder = 1, Rape = 2, Homicide = 3, and so on; the values of the eyes-color attribute can be encoded so that Black=1, Blue=2, Brown=3, Green=4, and in the same manner for all attributes with alphanumeric values.

Assume that each entity and each query, represented by numeric vectors only, are structured into an $m \times n$ matrix called E and into an $\ell \times n$ matrix called Q, respectively, and that the n attributes are *single-value* attributes *only*. To perform a *search* becomes a simple task. It requires determining whether the number of identical elements, when comparing corresponding elements of an entity vector and a query vector, is equal to the number of non-zero elements specified in the query vector. *If the answer is positive*, it indicates that the respective entity is represented by a point found on the searching hyperplane. Algebraically, if $E=[e_{ij}]$ and $Q=[q_{kj}]$ are the two matrices, and denoting the comparison of identical elements $(e_{ij}*q_{kj})=1$ and non-identical elements $(e_{ij}*q_{kj})=0$, the above condition is met if and only if (iff)

$$\sum_{j=1}^{n} (e_{ij}*q_{kj}) = t_k \quad \text{where } t_k = n - z_k \text{ and } t_k \leq n$$

z_k being the total number of zero elements specified in the k query vector.

The total search process can therefore be performed by a *Generalized Inner Product* (GIP) between matrices E and Q. The GIP permits the use of any *dyadic* operator between corresponding elements of two matrices where the only necessary condition is, as usual, that the two matrices must be *conformable* (the number of columns of the left matrix must be equal to the number of rows of the right matrix). Two among many acceptable GIP forms (in the APL

language) are:

$$A+.\times B; \text{ and } A+.=B$$

The first form is the known matrix product, the last form is the one used to perform the search described above.

By "post-multiplying" (using the GIP+.=) the entity matrix E by the transpose of the query matrix Q a response matrix R of $m\times\ell$ dimensions is obtained. Denoting by Q^T the transpose of Q

$$R=E+.=Q^T \quad r_{ik}=\sum_{j=1}^{n}(e_{ij}=q_{jk}) \quad \text{and } r_{ik}\le t_k$$

where $i=1,2,\ldots,m$ $j=1,2,\ldots,n$ $k=1,2,\ldots,\ell$. The point representing entity i is found on the searching hyperplane defined by query vector k iff $r_{ik}=t_k$. Clearly, each column of the R matrix contains the answer to the respective ℓ queries. If $r_{ik}<t_k$ the indication is that entity i has only r_{ik} identical properties with those specified in the query vector k. According to the user's needs entities can be retrieved for any r_{ik} value.

MULTI-DIMENSIONAL PROPERTY-SPACE MODEL

To illustrate the strength of the model developed and described in this section, a specific case must be analyzed first. In the section that describes queries and searching hyperplanes in the RDO model, the Range Query was discussed. This is one type of query that is most frequently used in searching criminal files. A witness may describe a criminal in the following terms: Age between 18 to 24, Height between 5.07 to 5.11, Weight between 170 to 190, Male, Caucasian. The ranges specified for the Age, Height and Weight attributes indicate that there is a certain *Degree of Uncertainty* (DOU). Assume that age is stored in increments of one year, the height in increments of one inch and the weight in increments of 10 lbs., and the DOU for each attribute is defined as the number of nonzero values within the respective specified ranges *minus* one. (The minus one comes to indicate that at least one value has to be always specified.)

In the given example, the DOUs are 6, 4 and 2, respectively. As explained previously, each range causes the query to be split. In this particular case, the query has a QCD=7+5+3=15; theoretically, at least 15 queries are needed to retrieve the set of criminals that qualify. The following process has to be invoked: (1) Structuring a 5-D subschema of attributes Age, Height, Weight, Sex, Race; (2) Formulating three 5-tuples (0,0,170,M,Cauc.) (0,0,180,M,Cauc.) (0,0,190,M,Cauc.); (3) Retrieving a subset of entities obtained from the union of the three previously retrieved subsets; (4) Searching the retrieved subset by five 5-tuples of the form $(0,V_2,0,0,0)$ where V_2=5.07, 5.08, . . . ,5.11; (5) Retrieving a subset of entities obtained from the union of the five previously retrieved subsets; (6) Searching the last retrieved subset by seven 5-tuples of the form $(V_1,0,0,0,0)$ where V_1=18,19, . . . ,24; (7) Obtaining the subset of entities that represent the answer, from the union of the seven previously retrieved subsets.

The same result can be obtained by formulating 105 5-tuples queries (c=7×5×3=105) of the form $(V_1,V_2,V_3,M,Cauc.)$ obtained from the Cartesian product $V_1\times V_2\times V_3$ where V_1={18,19, . . . ,24}; V_2={5.07,5.08, . . . ,5.11} and V_3={170,180,190}. The final subset that represents the answer is obtained from the union of the 105 previously retrieved subsets.

Despite the use of computers where the union, intersection and all other operations can be easily performed on large number of sets, the approach just described for handling one request appears to be too complicated, more so when many similar requests have to be satisfied over a short period of time. The solution to this problem, to problems previously described and to additional stringent requirements is found in the multi-dimensional property-space model, represented by the extended binary vectors and the binary connection matrices.

THE BINARY CONNECTION MATRIX

An n-D attribute-space containing $A_1,A_2,\ldots,A_n$ attributes, with their domains of distinct elements $D_1,D_2,\ldots,D_n$ of cardinality $d_1,d_2,\ldots,d_n$, respectively, can be transformed into an N-dimensional property space where $N=\sum_{i=1}^{n}d_i$ represents the number of distinct properties $P_1,P_2,\ldots,P_N$ necessary to map any point from the n-D attribute-space into the N-D property-space. Obviously N is larger than n, and to represent a point in a multi-dimensional Euclidean space requires many more coordinate axes, thus larger vectors. In the property space, however, only two distinct points exist on each axis; *zero* and *one*, and each coordinate axis represents a specific property.

An entity that is described in the attribute-space by n single-value attributes; i.e., by an n-tuple, is described in the property space by a binary N-tuple with n *one* values and N-n zero values. The *one* values are inserted in the positions representing the applicable properties. Since the cardinal number of attribute Sex is two (M,F) and the cardinal number of attribute Eyes-color is five (Black,Blue,Brown,Green,Hazel) the entity with properties (M,Blue); i.e., Sex-M and Eyes-Blue is represented by the 7-tuple E(1,0,0,1,0,0,0). The 2-tuple is transformed into a 7-tuple with *two* nonzero elements, and if the single value attributes would have respective cardinal numbers, say, d_1=10 and d_2=12 the 2-tuple from the attribute-space would be transformed into a 22-tuple in the property-space, again a binary vector with *two* nonzero values but 20 zero values. These vectors of the property-space are called *Extended Binary Vectors* (EBV) and they are usually very sparse.

A set of m entities will be described by a $m\times N$ binary matrix called the *Binary Connection Matrix* (BCM), because its nonzero elements indicate the connection that exists between each entity and its respective applicable properties, it is more specifically called the *Binary Property Matrix* (BPM). One important feature of EBV is that single-value attributes and multi-value attributes are represented by one and the same vector solving the serious redundancy

problem described previously. The 6-*D* attribute-space (the name-attribute is considered a unique identifier) of the criminal file from Figure 1, is transformed into 32-*D* property-space as illustrated in Figure 2. The corresponding BPM is illustrated in Figure 3. Obviously, in this matrix numeric and alphanumeric values are represented in the same manner, there is no need for special encoding as in the RDO model since the nonzero bit position implies the value.

The matrix is very sparse and this is another advantage because the application of sophisticated sparse matrix techniques not only provides the capability to save in storage requirements (only the nonzero elements are stored) but it provides the means to perform very efficient operations on the matrices (only the nonzero elements are operated on) as indicated in References 4 through 7.

Among other advantages the binary matrices enable one to use *Boolean Algebra* algorithms to solve data base problems (see References 8 and 9), to eliminate the difficulties of the range-query, to gather statistical data, to determine the relationships among all entity pairs and all property pairs, to assure privacy and security and many more.

Just looking at the binary matrix of Figure 2, one can scan the rows and see the applicable properties of each and every entity, but scanning the columns enables one to determine all entities that have a specific property. This is the most important feature of the *fully inverted files*, where for each and every property, a list of entities possessing the property is maintained. Therefore, the BCM in the property-space represents both a direct file and a fully inverted file at the same time, which is a very significant feature, to be fully taken advantage of in this investigation.

SEARCHING THE PROPERTY-SPACE

The search in this model is performed much in the same manner as the search in the attribute-space model. The query vector must have the same dimensionality as the entity vector, it indicates the coordinates of the projection

		CRIMES							ARM TYPE					LOCATIONS						SEX		EYE COLOR					LANGUAGES						
		Murder	Homicide	Kidnapping	Rape	Hijacking	Drug Push.	Armed Rob.	A	B	C	D	E	Chicago	Detroit	Houston	Los Angeles	New York	Washington	M	F	Black	Blue	Brown	Green	Hazel	English	French	German	Italian	Russian	Spanish	Swedish
ID	NAME	1	2	3	4	5	6	7	1	2	3	4	5	1	2	3	4	5	6	1	2	1	2	3	4	5	1	2	3	4	5	6	7
1	John	1							1						1					1			1				1						
2	Claude				1			1		1								1		1					1		1	1	1				
3	Robert					1						1				1				1						1	1		1				
4	Nancy		1						1					1							1		1				1		1				
5	Ingmar	1			1							1			1					1			1				1				1		1
6	Roberto	1						1						1						1		1					1			1		1	
7	Marcel			1		1					1							1		1					1			1	1			1	
8	Johanna						1	1					1				1				1				1		1	1				1	
9	Jurgen		1	1						1					1								1				1		1		1		
10	Tom						1	1			1						1			1					1		1	1					1
		1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10	1	2
																					20										30		

*(Crime Location)

Figure 2—The binary property matrix of the RDO sample

```
        1  0⍕CR
10000001000001000010001001000000
00010010100000001010000101110000
00001000001000100010000011010000
01000001000010000001001001010000
10010000001001000010001001000101
10000010000110000010100001001010
00101000010000001010000100110010
00000110000100010001000101100010
01100000100001000010001001010100
00000110010000100100000011100001
```

Figure 3—The binary property matrix (compact form)

point, on the hyperplane defined by its nonzero values, that all entity points must have in order to qualify. Denoting the $m \times N$ entity matrix by E and the $\ell \times N$ query matrix by Q, the search process is performed again by a generalized inner product (GIP) between the $E=[e_{ij}]$ matrix and the transpose of the $Q=[q_{kj}]$ matrix named Q^T. By post multiplying E by Q^T the $m \times \ell$ response matrix $R=[r_{ik}]$ is obtained. This time, however, a regular matrix multiplication is performed, because the search criterion requires determining the number of corresponding nonzero elements between the entity vector and the query vector (i.e., the number of identical properties specified in the two vectors) and, for binary vectors

$$e_{ij} \times q_{kj} = 1 \text{ iff } e_{ij}=1 \text{ and } q_{kj}=1;$$

$$\text{for other cases } e_{ij}=q_{kj}=0$$

$$R=E+.\times Q^T$$

$$r_{ik}=\sum_{j=1}^{N} e_{ij} \times q_{kj}$$

$$\text{where } r_{ik} \leq t_k$$

$$t_k=\sum_{j=1}^{N} q_{kj}$$

The number of nonzero values specified in a vector is called the *weight of the vector*, the weight of vector Q_k is t_k. The matrices in Figure 4 illustrate the searching process.

The *Negation Query* is another efficient form of file searching. Rather than specifying in the query the properties that an entity must have, the *one* values in the query indicate the properties that the entities must *not* have. For example, the hijacker does not speak English, French and German. By post "multiplying" E by Q^T using a GIP form

$$R=E+.<Q^T \text{ which is}$$

$$r_{ik}=\sum_{j=1}^{N} (e_{ij}<q_{kj}) \quad r_{ik} \leq t_k \quad t_k=\sum_{j=1}^{N} q_{kj}$$

The Range Query. Let attribute Age have six distinct values; 20,21, . . . ,25 and attribute Weight, four distinct values; 160,170,180,190. In the property-space the Age,Weight attributes for m entities are represented by m 10-D binary vectors representing the respective property positions

P	(20,	21,	22,	23,	24,	25,	160,	170,	180,	190)
E_1	(0,	0,	1,	0,	0,	0,	0,	1,	0,	0)
Q_1	(0,	1,	1,	1,	1,	0,	0,	1,	1,	1)

The E_1 entity vector represents a criminal of Age-22 and Weight-170, all other $(m-1)$ entity vectors have *two* nonzero values at different positions. The Q_1 query vector represents the range query for the identification of all criminals of age between 21 to 24 and weight 170 to 190. By post multiplying the $m \times 10$ entity matrix by the transpose of the Q_1 query vector, an $m \times 1$ R_1 response vector is obtained. The elements R_{i1}; $i=1,2, \ldots, m$ of the response vector have values of 0, 1, or 2. Clearly, only entities associated with $R_{i1}=2$ qualify, that $R_{11}=2$ is obvious. As seen in this particular case, the weight of the query vector is $t_1=7$ and $r_{i1} \leq 2$, the cause is selfexplanatory.

Thus, this is the simple and elegant solution to the complicated range-query problem postulated at the beginning of a former section. Instead of 15 queries and the necessary set-operations, or instead of the 105 queries with their set-operation, one and *only* one query vector with a weight $t_k=17$ will produce a response vector with elements $r_{ik} \leq 5$, and all the entities with $r_{ik}=5$ will qualify. This is illustrated in Figure 5, where a number of range-queries searched simultaneously the entity set.

The OR-Query. In the RDO model the OR operator always causes a split of the query, in the BPM model it

```
        1  0⍕Q
10000001000001000010001001000000
00001000001000100010000011010000
10010000001001000010001001000101
00000110000100010001000101100010
00000110010000100100000011100001

        TX←+/Q
        TX
6  7  9  9  9

        RQ←CR+.×⍉Q
        RQ
 6  2  5  1  2
 2  3  3  4  4
 2  7  3  1  3
 3  2  2  2  1
 5  3  9  1  3
 3  2  3  4  3
 1  3  1  3  3
 1  1  1  9  5
 4  3  5  1  2
 2  3  3  5  9
```

Figure 4—The searching process, query and response matrices

```
ABWSRB IS A BINARY MATRIX PRODUCED BY THE LAMINATION OF AGE HGT, WGT,SEX AND RACE
Q10 IS A BINARY QUERY MATRIX THAT CONTAINS FIVE RANGE QUERIES
GIL IS A FUNCTION THAT IDENTIFIES ENTITIES THAT QUALIFY FROM THE RESPONSE MATRIX
THE LEFT ARGUMENT OF GIL CONTAINS THRESHOLD VALUES FOR THE RESPECTIVE QUERIES
COLUMNS 1-32 REPRESENT THE AGES 18 YRS.-50 YRS.
COLUMNS 33-51 REPRESENT HEIGHTS 60 INCH.-78 INCH.
COLUMNS 52-67 REPRESENT WEIGHTS 100 LBS.-250 LBS.
COLUMNS 68-69 REPRESENT THE SEX M AND F RESPECTIVELY
COLUMNS 70-73 REPRESENT RACES CAUC.; EURAS. NEGRO ORIEN.

      1 0⍉ABWSRB[⍳4;]
|00000100000000000000000000000000|0000000000100000000|0000000010000000|10|0001|
|00000000000000000100000000000000|0000000000010000000|0000000100000000|10|1000|
|00000000010000000000000000000000|0000000000000010000|0000000000001000|10|0001|
|01000000000000000000000000000000|0000100000000000000|0001000000000000|01|1000|
      1 0⍉Q10
|11111100000000000000000000000000|0000000000110000000|0000000111000000|10|1000|
|00011111110000111100000000000000|0000001111111111000|0000111111111000|10|1100|
|00000000000000000000000000000000|1111111000000000000|1111100000000000|01|1000|
|00000000000000000000000001111100|0000000000000000000|0011111000000000|10|1111|
|11111111111111110000000000000000|1111111000000000000|0001111111110000|11|1000|
      R10←ABWSRB+.∧⍉Q10
      RE10←5 5 4 4 5 GIL R10
      RE10[;⍳24]
 9  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 2 12 17 23 27  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 4 20  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 4 20 23  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

Figure 5A—Search by range queries

depends on the form of the request. To identify all criminals of age 27, height 5'10" and blue or green or hazel eyes requires the formulation of one query vector, of weight $t_q=5$; the entities that qualify have an $r_{eq}=3$. The same request, only this time for criminals of age 27 or 28 or 29, will require again one query of the same dimensionality with a $t_q=7$ and $r_{eq}=3$. Actually, these *OR* queries are range queries. But, if the request is age 27, height 5'10" and blue or green eyes, or age 28, height 5'10" and green or hazel eyes, or age 29, height 5'10" and blue or hazel eyes, three separate queries are required.

Denoting attributes Age, Height and Eyes Color by *A*, *H* and *C*, respectively, and using switching theory notation, the three requests can be formulated as

(1) $A_1H_1(C_1+C_2+C_3)$
$=A_1H_1C_1+A_1H_1C_2+A_1H_1C_3 \rightarrow 3$ triples

(2) $(A_1+A_2+A_3)H_1(C_1+C_2+C_3)$
$=A_1H_1C_1+A_1H_1C_2+\ldots+A_3H_1C_3 \rightarrow 9$ triples

(3) $A_1H_1(C_1+C_2)+A_2H_1(C_2+C_3)+A_3H_1(C_1+C_3)$
$=A_1H_1C_1+A_1H_1C_2+\ldots+A_3H_1C_3 \rightarrow 6$ triples

Obviously, all entities that can be projected in any of the three points specified by the triples of the first request will qualify for the first, second and third request, all the entities that can be projected in any of the six points specified by the third request will qualify for the second and third requests but not all entities that qualify for the second request will qualify for the first and third requests. In the property-space, *m* entities can be represented by a sub-schema that contains all the properties specified in the three requests by *m*, 7-*D* binary vectors representing the respective property positions associated with each entity. Each and every entity can have a maximum of three nonzero values, because Age, Height and Eyes Color are single-value attributes.

$$
\begin{array}{l}
P(A_1, A_2, A_3, H_1, C_1, C_2, C_3) \\
E_1(1,\ 0,\ 0,\ 1,\ 0,\ 0,\ 1) \\
Q_1(1,\ 0,\ 0,\ 1,\ 1,\ 1,\ 1) \\
Q_2(1,\ 1,\ 1,\ 1,\ 1,\ 1,\ 1) \\
Q_3'(1,\ 0,\ 0,\ 1,\ 1,\ 1,\ 0) \\
Q_3''(0,\ 1,\ 0,\ 1,\ 0,\ 1,\ 1) \\
Q_3'''(0,\ 0,\ 1,\ 1,\ 1,\ 0,\ 1)
\end{array}
$$

It is apparent, however, that while the Q_1 query vector defines the three necessary triples and the Q_2 query vector defines the nine necessary triples, to define the six necessary triples for the third request three separate queries Q_3', Q_3'' and Q_3''' have to be formulated. This is required in order to exclude the three underscored triples from the nine

$$
\begin{array}{l}
A_1H_1C_1; A_1H_1C_2; \underline{A_1H_1C_3} \\
\underline{A_2H_1C_1}; A_2H_1C_2; A_2H_1C_3 \\
A_3H_1C_1; \underline{A_3H_1C_2}; A_3H_1C_3
\end{array}
$$

possible triples. The response to the third request is obtained by taking the *union* of the three subsets retrieved by

$Q_3'Q_3'$ and Q_3''. Observe that entity E_1 that possesses the projection $\underline{A_1H_1C_y}$ qualifies for the first and second requests but does not qualify for the third. Informally, it may be stated that the number of separate queries to be formulated is equal to the number of members in the left-hand side of the equation.

HUMAN-FACE IDENTIFICATION

Human-face features of a large population can be stored in a computer. The problem is how to structure the data so that an efficient search can be performed to identify an individual when a number of features is specified, or how to

```
THE IDENTIFIERS FROM REL0 ARE USED IN THE FOLLOWING TABLES

      V1
ID AGE HEIGHT WEIGHT SEX
      T1
IDG[I;] AGG[I;] HEG[I;] WEG[I;] SEXG[I;]

      I←REL0[1;1]

      V1 TABLE T1
  ID  AGE  HEIGHT  WEIGHT  SEX
   9   18    5.11     190   M


      V10
ID NAME AGE HEIGHT WEIGHT SEX RACE
      T10
IDG[I;] NAMG[I;] AGG[I;] HEG[I;] WEG[I;] SEXG[I;] RACEG[I;]

      I←REL0[2;1 5]

      V10 TABLE T10
  ID      NAME  AGE  HEIGHT  WEIGHT  SEX       RACE
   2    CLAUDE   35    5.11     170    M  CAUCASIAN
  12      MARK   24    5.06     150    M   EURASIAN
  17      RUDY   25    6.03     210    M  CAUCASIAN
  23    GERALD   32    5.06     160    M  CAUCASIAN
  27    DONALD   26    6.01     220    M   EURASIAN


      I←REL0[3;1 2]

      V10 TABLE T10
  ID      NAME  AGE  HEIGHT  WEIGHT  SEX       RACE
   4     NANCY   19    5.04     130    F  CAUCASIAN
  20     JULIA   24    5.05     140    F  CAUCASIAN


      I←REL0[5;1 3]

      V10 TABLE T10
  ID      NAME  AGE  HEIGHT  WEIGHT  SEX       RACE
   4     NANCY   19    5.04     130    F  CAUCASIAN
  20     JULIA   24    5.05     140    F  CAUCASIAN
  23    GERALD   32    5.06     160    M  CAUCASIAN
```

Figure 3B—Retrieval after range query search

classify individuals in distinct classes as a function of their features. A few researchers have investigated the problem.[illegible] It appears, however, that more efficient methods are needed to obtain a quick answer, especially when many different descriptions are to be considered simultaneously.

Structuring the data in a binary property-matrix form, as illustrated in Figure 6, and Figure 7, provides the means for fast identification and classification. The tables are self-explanatory, the attributes can be permuted in any order so that many different sub-schemas can be derived. If a witness describes a criminal that had his face covered from the nose down, only the hair, eyebrows, eyes, forehead and ears are needed in the sub-schema. A witness may describe a criminal without being able to assign a specific value to each and every feature; in other words, features may be described with different degrees of uncertainty (DOU). For example, describing the nose, the witness may say that he is not sure if the nose-length should be considered as short or medium and if the profile is concave, straight, or somewhere inbetween. In this particular case, these attributes will be covered by a *range-query* of the form (1,1,1,0,0,1,1,1,0,0).

The strength of the method is not restricted to the capability of applying range-queries to each and every attribute at once, so as to overcome the uncertainty problem. Assume that several witnesses describe the same criminal, or a number of criminals, with different DOUs, the process of interrogating the entity set with many queries concurrently, called a parallel search, provides the capability to check a number of alternatives, as illustrated in Figure 8.

Distinctive features are easily discovered in this model by calculating a figure of merit called the *Degree of Distinction* (DOD). Assume that an $m \times N$ matrix $E=[e_{ij}]$ contains the data of m human faces, some of the N features are associated with fewer faces than others, these are the more distinctive features. The DOD for each and every property can be calculated by dividing the column vector weights of the E matrix by m,

$$DOD(j)=\frac{1}{m}\sum_{i=1}^{m} e_{ij}$$

The lower DOD(j) values indicate the most distinctive properties P_j, which are very important indicators in determining the features to be described in a query to get the most pertinent answer.

The human face file contains n single-value attributes (eyes-opening, eyes-separation, eyes-color are three distinct attributes) and considering that the row vector weights of an n-attribute entity set is $\sum_{j=1}^{N} e_{ij}=n$, the degree of uncertainty reflected by each query vector of a $Q=[q_{kj}]$ query matrix that interrogates the $E=[e_{ij}]$ entity matrix can

	FOREHEAD					CHEEKS					CHIN					MOUTH							
											Profile					Width					Lip*		
	Receding		Vertical		Bulging	Sunken		Average		Full	Receding		Straight		Jutting	Small		Medium		Large	Upper	Neither	Lower
ID	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	1	2	3
1	1									1		1								1	1		
2		1					1								1		1						1
3				1					1					1					1		1		
4			1					1					1			1						1	
5					1	1						1						1					1
6				1					1						1					1		1	
7		1					1				1					1						1	
8			1			1							1					1			1		
9	1							1			1								1			1	
10					1					1					1		1						1
	4	5	6	7	8	9	0 40	1	2	3	4	5	6	7	8	9	0 50	1	2	3	4	5	6

	NOSE													EARS									
	Length					Profile					Tip			Length					Protrusion				
	Short		Medium		Long	Concave		Straight		Hooked	Upward	Horizon	Downward	Short		Medium		Long	Slight		Medium		Large
ID	1	2	3	4	5	1	2	3	4	5	1	2	3	1	2	3	4	5	1	2	3	4	5
1		1						1				1					1				1		
2					1					1			1		1								1
3			1					1				1						1	1				
4	1					1					1			1		1						1	
5				1					1			1					1			1			
6				1			1				1			1						1			
7		1							1				1		1				1				
8			1					1				1				1					1		
9					1	1					1						1					1	
10					1					1			1					1					1
	7	8	9	0 60	1	2	3	4	5	6	7	8	9	0 70	1	2	3	4	5	6	7	8	9

*(Lip Overlap)

Figure 6—Additional features of human-faces

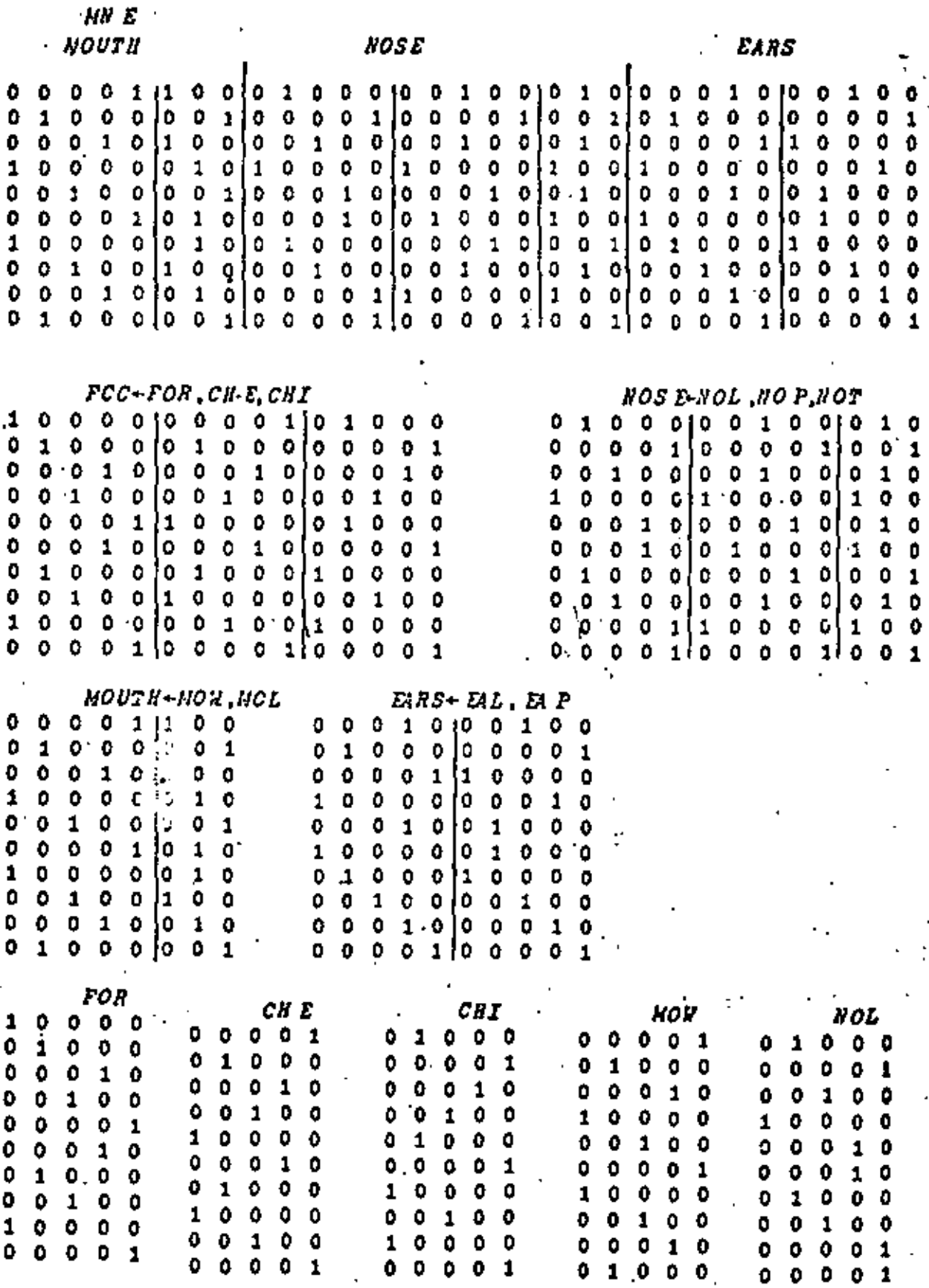


Figure 3—Binary matrices of human-face features

```
        QFCC
 0  0  0  0  0  1  1  0  0  0  0  0  1  1  1
 0  1  1  1  0  0  0  1  1  1  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  1  1  0
 0  0  0  1  1  0  1  1  0  0  0  1  1  1  0
 1  1  0  0  0  1  1  1  0  0  0  0  0  1  1
 0  0  1  1  1  0  0  1  0  0  1  1  1  0  0
 1  0  0  0  0  0  0  1  0  0  1  1  0  0  0
 1  1  0  0  1  1  0  0  0  0  0  0  0  1  0
        TX
 5 6 2 7 7 7 4 4
        VK
 2 2 1 3 3 3 3 3

        DOUK←TX-VK
        DOUK
 3 4 1 4 4 4 1 1

        RFCC←FCC+.×⍉QFCC
        TRFCC←V GIL RFCC
        TRF←TRFCC[;⍳12]

        TRF
  2   8  11  14  28  29   0   0   0   0   0   0
  3   4   6  15  19  22  23  25  26  27  30   0
  3   4   8  12  15  23  25  26  28   0   0   0
 23  30   0   0   0   0   0   0   0   0   0   0
  2  27   0   0   0   0   0   0   0   0   0   0
  4  23  30   0   0   0   0   0   0   0   0   0
  9  24   0   0   0   0   0   0   0   0   0   0
  0   0   0   0   0   0   0   0   0   0   0   0

        VAL3
ID NAME CRIME LOCATION DATE DETECTIVE
        TAG3
IDC[I2;]NAMC[I2;]CRC[I2;]LOCC[I2;]DATEC[I2;]DETC[I2;]
        I2←TRF[6;⍳3]

        VAL3 TABLE TAG3

 ID     NAME        CRIME       LOCATION        DATE      DETECTIVE
  4     NANCY       HOMICIDE    DETROIT       03/15/73    GONZALEZ
 23     GERALD      ARMED ROB.  DETROIT       12/25/75    BURNSIDE
 30     JEAN        ARMED ROB.  LOS ANG.      12/25/72    EXPOSITO

        DODJ←(+/FCC)÷1↑⍴FCC
        4 2⍕DODJ
 .13 .17 .23 .27 .20 .20 .23 .23 .17 .17 .23 .20 .27 .03 .27
```

Figure 8—Search and retrieval by queries with different DOUs

```
MDOD CONTAINS THE 10 MOST DISTINCTIVE FEATURES OF MNE

        DOD←DIAG FS                 ES
        ρDOD
31
        MD←⍋DOD

        MDOD←MD[⍳10]
        MDOD
18 17 23 24 1 4 13 27 28 31

        DOD[MDOD]
3 4 4 4 5 5 5 5 5 5

        DMNE←MNE[;MDOD]

        +/DMNE
3 4 4 4 5 5 5 5 5 5

        +/DMNE
0 4 2 1 2 1 4 1 2 3 2 1 2 1 1 0 1 1 0 0 1 1 2 3 2 0 1 2 2 2

        FSD←(⍉DMNE)+.×DMNE
        FSD
 3 0 1 0 0 0 2 0 0 3
 0 4 1 0 1 0 1 2 1 0
 1 1 4 0 1 1 1 1 1 1
 0 0 0 4 1 0 0 1 0 0
 0 1 1 1 5 0 0 1 0 1
 0 0 1 0 0 5 2 1 0 0
 2 1 1 0 0 2 5 1 0 2
 0 2 1 1 1 1 1 5 0 0
 0 1 1 0 0 0 0 0 5 0
 3 0 1 0 1 0 2 0 0 5

        DIAG FSD
3 4 4 4 5 5 5 5 5 5

        I←3 IGE +/DMNE
        I
2 7 10 24

        VAL4 TABLE TAG4

        NAME     AGE  HEIGHT   WEIGHT   EYES    PROF.       CITY
    CLAUDE       35    5.11      120   GREEN   B.KEEPER    HOUSTON
    MARCEL       42    5.08      180   GREEN   ACTOR       NEW YORK
    TOM          44    6.00      200   BLUE    MECHANIC    SAN FRAN.
    MARGOT       19    5.09      180   BLACK   SECRETARY   DETROIT
```

Figure 9—Similarity matrix of ten distinctive features

be determined as follows

$$DOU(i)=\sum_{j=1}^{n} q_{ij}-V(i) \quad \text{where } V(i)\leq n$$

The $V(i)$ value is determined by the number of attributes covered by query Q_i. The two defined coefficients DOD and DOU can be calculated in many ways as illustrated in Figure 8, but they are readily obtained from Similarity Matrices.

The Entity Similarity Matrix (ES) is a symmetric $n\times n$ matrix obtained by post-multiplying the entity matrix E by its transpose

$$ES=E+.\times E^T=[e_{ik}]\times[e_{jk}] \quad es_{ij}=\sum_{k=1}^{n} e_{ik}e_{jk}$$

where $i=1,2,\ldots,n$; $k=1,2,\ldots,m$.

The (i,j)th element of ES represents the number of common properties possessed by the pair of entities E_i and E_j. When $i=j$ the elements (i,i) of ES are the elements of the main diagonal, each element representing the number of properties specified in entity E_i, which is the weight of vector E_i. The (i,j) elements indicate the degree of similarity between entities E_i and E_j.

The Property Similarity Matrix (PS) is obtained by pre-multiplying the E matrix by its transpose E^T, it is an $N\times N$ symmetric matrix $PS=E^T+.\times E$. The (i,j)th element of PS represents the number of entities that possess the properties P_i *and* P_j. The (i,i)th element of the main diagonal represents the number of entities associated with property P_i. Dividing the (i,j) elements by n the $DOD(i)$ is obtained.

The Query Similarity Matrix (QS) is obtained by post-multiplying the query matrix Q by its transpose Q^T, if Q is an $\ell\times N$ matrix, QS is a symmetric $\ell\times\ell$ matrix. The (i,j)th element of QS represents the number of common properties specified in queries Q_i and Q_j. The (i,i)th element of the main diagonal represents the number of properties specified in query Q_i, this number was defined, previously, as the query complexity degree QCD. Subtracting from the element of the main diagonal the $V(i)$ value, which represents the number of attributes covered by query Q_i, the $DOU(i)$ is obtained.

The $V(i)$ value is called the *Covering Coefficient of* query Q_i and as stated previously $V(i)\leq n$. If there are n attributes it is known that $N=\sum_{i=1}^{n} d_i$ is the dimensionality of the query vector in the property-space. One can, however, specify in a Q_i query vector of dimensionality N, properties pertaining to k domains ($k\leq n$); i.e., the query *covers* only k attributes, therefore $V(i)=k$.

The Query-Property Similarity Matrix (QPS) is an $N\times N$ symmetric matrix obtained by pre-multiplying the Q matrix by its transpose Q^T. The (i,j) element of QPS indicates the number of queries in matrix Q in which properties P_i *and* P_j are specified, this is a *Pair-Usage-Frequency Indicator* (PUFI) because it indicates how frequently pairs of properties are specified in queries. The (i,i) element of QPS indicates the number of queries in matrix Q in which property P_i is specified, it is called the Usage Frequency Indicator (UFI). The UFI is a very important factor in determining the best strategy for storage allocation.

One of the most important aspects of this associative approach is its great flexibility in *Logical Reconfiguration*; i.e., the capability to "cut and paste" attributes or properties as a function of the time-variable queries. As illustrated in Figure 7 by the matrices FOR; CHI;; CHI; MOW; MOL; NOL; NOP, etc., one can concentrate on the forehead, cheeks, chin, mouth's width, mouth's lip overlap, nose's length, nose's profile, respectively, or any combination thereof, to create new subschemas or *Aggregates*. If a witness recalls that a criminal has a long nose and curly hair, the search can be performed over attributes hair-texture and nose-length, or over any specific properties. The flexibility of this approach enables one to create aggregates of properties that belong to different attributes. One can also create aggregates of the most distinctive features stored in a file, for example, the ten properties with the lowest DOD values as illustrated in Figure 9.

If the intersection of the set of criminals that possess the most distinctive properties with the set of criminals retrieved by the search over the properties specified by the witness is not empty, the interrogator may ask the witness to try and recall if the criminal did not have any of the most distinctive features.

CONCLUSIONS

Only a few basic concepts of SPARCOM have been presented in this paper. Many other concepts and algorithms have also been implemented and tested. The APL is used here merely as a tool to explain the concepts. The strength of the method and the efficiency of the system have been clearly demonstrated in an Assembler Language implementation when a Data Base of 100,000 entities vs. 65,000 potential properties was searched concurrently by 30 complex queries. (Each query contained over 30 different properties from a set of 65,000 potential properties.) On a system 360 Mod 85 all entities that qualified were retrieved within 60 seconds. A detailed description of SPARCOM's concepts can be found in References 7 and 12.

REFERENCES

1. Codd, E. F., "A Relational Model for Data for Large Shared Data Banks," *Comm. ACM* 13, 6, June 1970, pp. 377-387.
2. Date, C. J., *An Introduction to Data Base Systems*, Addison-Wesley Pub. Co., Reading, Massachusetts, 1975.
3. Martin, J., *Computer Data-Base Organization*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1975.
4. Ashany, R., "Sparse Matrix Techniques for Information Storage and Retrieval," *Proc. of Fifth Annual Princeton Conf. on Inf. Sciences and Systems*, Princeton, N.J., March, 1971.
5. Ashany, R., "Automatic Classification and Relationship Analysis for Information Storage and Retrieval," *Proc. of Third Southeastern Conf. on System Theory*, Atlanta, Georgia, April, 1971.
6. Ashany, R., "Concepts of Data Manipulation—The Connective Matrix Method," IBM Technical Report T. R. 00.2300, Poughkeepsie, N.Y., June, 1971.

7. Ashany, R., "SPARCOM: A Sparse Matrix Associative Relational Approach to Dynamic Data Structuring and Data Retrieval," IBM Technical Report T. R. 00.2774 Poughkeepsie, N.Y., July 1976.
8. Delobel, C. and R. G. Casey, "Decomposition of Data Base and the Theory of Boolean Switching Functions," *IBM J. R&D* 17, 5, September 1973, pp. 374-386.
9. Delobel, C. and J. Rivenen, "Decomposition of Files, A Basis for Data Storage and Retrieval," IBM Research Report RJ1220, San Jose, Calif., May 10, 1973.
10. Goldstein, L. D., et al., "Man-Machine Interaction in Human-Face Identification," *The Bell System Technical Journal*, Vol. 51, No. 2, February 1972, pp. 399-427.
11. Goldstein, L. D. et al., "Identification of Human Faces," *Proc. IEEE*, 59, No. 5, 1971, pp. 748-760.
12. Ashany, R., "SPARCOM: A Sparse Matrix Associative Relational Approach to Dynamic Data Structuring and Data Retrieval," Ph.D. Dissertation, Dept. of Electrical Engineering, Polytechnic Institute of New York, June 1976.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

GO SYSTEM-DESIGN AND IMPLEMENTATION OF AN OUTPUT GENERATOR

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

GO System—Design and implementation of an output generator*

by ROLAND R. BONATO and KENNETH C. YANG
The George Washington University
Washington, D.C.

ABSTRACT

This paper concerns a conceptual model for a general purpose output generator and its implementation. The Generated Output (GO) System is based on two major criteria: (1) user oriented and (2) easy maintenance. Its goal is to produce tailored output in an unambiguous, camera-ready form. Basically, the system is comprised of a derived file, a descriptor file and an interface module which integrates the two files according to externally-supplied specifications. Derived data are usually numeric results such as percentages, totals, subtotals, etc. Descriptors are defined as alphanumeric labels whose function is to clearly identify derived data. Classification of different types of calculated and label files are discussed with accompanying examples and illustrations.

BACKGROUND

In a society where computer output is being widely used for business and scientific reporting, this medium has become an increasingly accepted vehicle for human communication. A review of computer output during the past two decades indicates that there is a trend toward more complex formatting. The change is mostly due to an expanding and more varied usership; i.e., as the consumer became more sophisticated, the demand for custom-tailored or even camera-ready output increased. For the most part, the DP community responded to these demands by developing a compiler approach to producing statistical tables. The Report Program Generator (RPG) by IBM and Table Producing Language (TPL) by the Bureau of Labor Statistics are examples of systems that were specifically designed to accommodate a variety of output displays.

INTRODUCTION

A major problem in the production of computer output involves the manipulation of alphanumeric descriptors, i.e., labels whose sole function is to clearly identify data formatted output. As opposed to calculations, labels are not readily derived from algorithmic models. At the Biometric Laboratory, The George Washington University, we reviewed the problems associated with output formatting and came to the following conclusions:

(1) Labeling requires extensive amounts of tedious formatting effort. Consequently, labeling tends to be sparse and abbreviated.
(2) Revision of output formats is equally tedious.
(3) General purpose programs, in order to maintain their generality, tend to resort to generic Labels, e.g. % or $. However, there are many instances where such generic labels do not suffice.
(4) When unambiguous labeling is required, the programmer resorts to "one shot" programming consisting of repetitive calculation logic and unique alphanumeric formats; i.e., the programmer is forced to "re-invent the wheel" calculationwise because of labeling specifics.

In order to reduce these problems and provide flexibility in generating output, we have developed a system based on two major criteria:

(1) User oriented—users should be able to specify not only numerical results but also the terminology to be employed and the positioning and ordering of their output.
(2) Easy maintenance—utilization and maintenance of the system should be accomplished mostly by trained technicians.

The following sections describe the design, basic concepts, implementation, and example output from our system.

DESIGN

Given the above-mentioned criteria, we decided to treat derived data and their associated descriptors as separate entities, i.e., independently-generated files.

* Preparation of this paper was supported by Grant #5-R01-MH-22019-03 from the National Institute of Mental Health, U.S. Public Health Service.

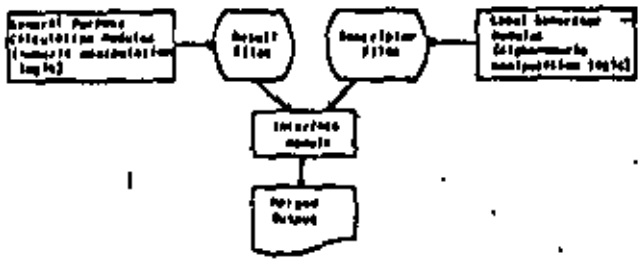


Figure 1—Design of an output generator

Derived data are viewed as calculated results such as percentages, totals, subtotals, etc. Descriptors are defined as those alphanumeric labels needed to identify results. Merged output is achieved via an interface module which accesses both the results and descriptor files as shown in Figure 1.

This design has the following advantages:

(1) General purpose calculation programs can be written independent of formatting constraints.
(2) The descriptor file can be modified and maintained as a separate entity.
(3) The interface module provides flexibility in the arrangement of final output, e.g., configuration of output can be tailored to the individual user without extensive reprogramming.

BASIC CONCEPTS

The purpose of the following discussion is to explore concepts that will yield results and descriptors with some degree of generality and a minimum of effort.

As viewed here, all output can be subdivided into a series of component parts.

Figure 2 represents the concepts involved in the GO system, from the most elementary to the most complex and their interrelationship. Starting at the top of the schematic, TOTAL OUTPUT is made up of PAGE IMAGES, and one or more TABLE(S) constitute a PAGE IMAGE. TABLES, in turn, are derived from RESULT CLASSES and LABEL TYPES. Finally, results and labels are composed of elements, i.e., a single calculation or an alphanumeric word.

A page image is defined as an output unit whose content forms a logical whole within the total output. Although the dimensions of a page image are theoretically unlimited, an actual page image introduces certain constraints; e.g., ma-

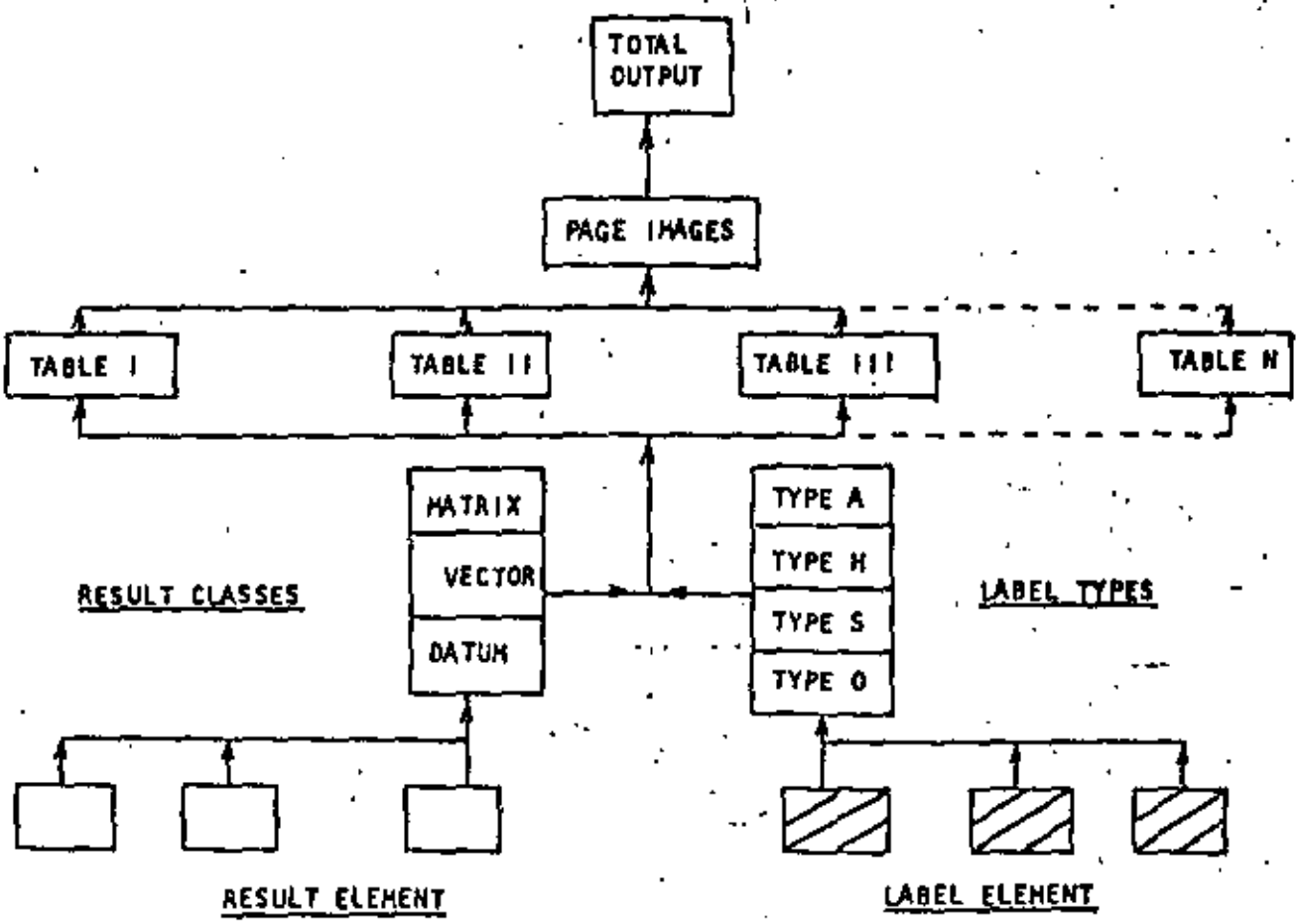


Figure 2—Components of output

chine limitations, such as finite number of print positions, tend to influence the page size.

A table is defined as a merged display of results and its descriptors or labels (whenever a single table is involved, the page image is the table). All results can be divided into three classes: a datum (1×1), a vector (1×*N*), and a matrix (*N*×*N*). Descriptors are divided into four types: All alphanumeric (TYPE A), Headers (TYPE H), Stubs (TYPE S) and Others (TYPE O).

Label definitions

(1) All alphanumeric (TYPE A) labels consist of one or more lines of pure alphanumeric output. As opposed to TYPES H and S, their position is minimally correlated with the result classes; i.e., the form and alignment of TYPE A is not determined by derived output. TYPE A is commonly used for table description, footnotes, etc.

(2) Headers (TYPE H) are defined as labels that describe columnar results. The position of TYPE H elements is a function of the format of the result matrix and vector.

(3) Stubs (TYPE S) are labels that define results rowwise and are similar to TYPE H in that the location of TYPE S elements is determined by results.

(4) Other (TYPE O) labels describe data plus special characters that are not necessarily associated with results elements.

Results definitions

The meaning of the three result classes is commonly known. However, the definitions apply to output requirements as opposed to the calculated results format.

Tables and page image

Usually, the most convenient type of output involves the juxtaposition of related data on a single physical page. This saves paging back and forth, transcribing, or cutting and pasting. In addition to convenience, interrelatedness and machine limitations, aesthetics may be a major determiner of the page image, e.g., neat and uncluttered output.

From the above, it becomes apparent that specification of page images is a function of the number, redundancy, symmetry, extent and variety of page elements. For example, when the same page image can be repeated across many physical pages (redundancy), the cost/benefit ratio is enhanced. Similarly, when label and result elements are evenly spaced, symmetrical tables are produced and the titioning of the table is simplified. Usually, the most asily" tables to specify are a series of non-redundant asymmetrical labels and results.

IMPLEMENTATION

An experimental model of GO (Generate Output) System has been developed and implemented at the Biometric Laboratory to test the concepts previously described. Throughout the developmental phase, we have designed numerous utilizable functions and implemented 3 principal modules and 9 sub-modules, serving key functions, for the GO System. All programs are written in FORTRAN and PL/I for an IBM 370/135—265K—OS/VS1 System.

The following represent the modules that have been implemented for the GO System:

(1) GOMAIN (Interface Main Module)—This module accepts input parameters, refers to appropriate result and label files, and combines appropriate submodules for flexible output. Its flexibility can be observed from the ensuing examples.

(2) GOCLAR (Interface sub-module)—To clear the area, buffer, in which the results and labels will be merged.

(3) GOREST (Interface sub-module)—To restore a page image for repetitious use of the same page image with differential results.

(4) GOGRED (Interface sub-module)—used as input function for generation of graph displays.

(5) GOGRPD (Interface sub-module)—used in conjunction with GOGRED to generate and plot data for displays.

(6) GODATM (Interface sub-module)—to transfer datum results to output buffer.

(7) GOVECT (Interface sub-module)—to transfer vector results to output buffer.

(8) GOMATX (Interface sub-module)—to transfer matrix results to output buffer.

(9) GOUPDT (Interface sub-module)—used as update/modify function for previously-created page images.

TABLE I—Page Image

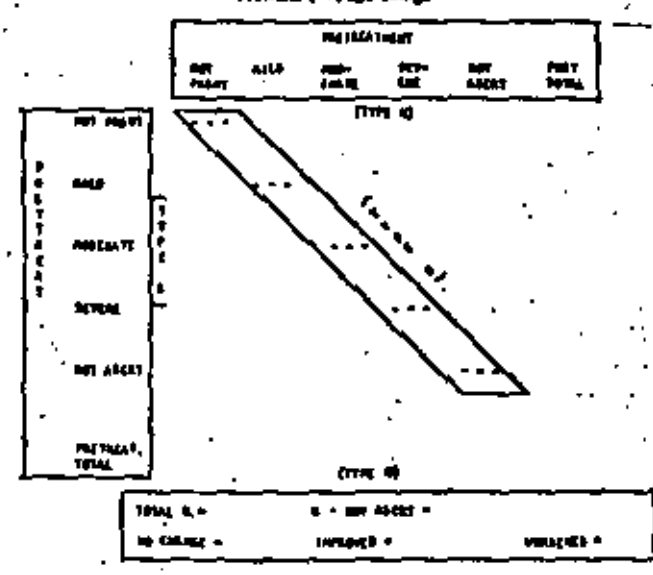


TABLE II—Merged Page Image and Results

(TYPE A) PSYCHIATRIC RATING SCALE - ANXIETY

PRETREATMENT

POSTTREAT	NOT PRSNT	MILD	MOD-ERATE	SEV-ERE	NOT ASCRT	POST TOTAL
NOT PRSNT	0	14	11	4	0	29
MILD	1	2	22	11	0	36
MODERATE	0	1	2	16	1	20
SEVERE	0	1	2	7	1	11
NOT ASCRT	0	0	1	0	0	1
PRETREAT. TOTAL	1	18	38	38	2	←VECTORS

MATRIX

TOTAL N = 97 → DATUM

NO CHANGE = 11 → DATUM

N - NOT ASCRT = 94

IMPROVED = 78 → DATUM

WORSENED = 5 → DATUM

(10) GOPRNT (Interface sub-module)—prints the merged output which resides in the buffer in user-specified format.

(11) GOPAGE (Label Generator Module)—to create Page Directory constituted of page images tailored to user-specified output.

(12) GOSTAB (General Purpose Calculation Module)—Multiways cross-tabulation and computations of n, ΣX, $\bar{X}$, and σ conducted (inclusive of row %, column %, table %, cumulative row %, cumulative column % for n and ΣY). The results are documented and stored on temporary disk file (result file) to be utilized by the GOMAIN.

During the developmental and design phase of the GO System, we also envisioned a spin-off from this concept.

TABLE III—Frequency Table

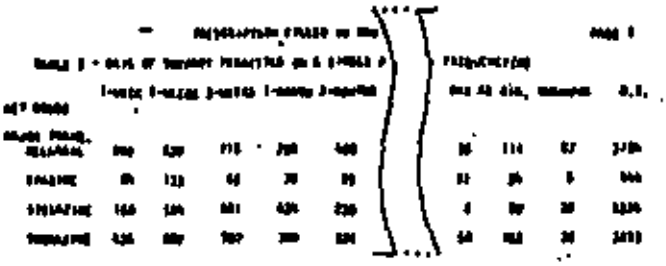


TABLE IV—Percent Table

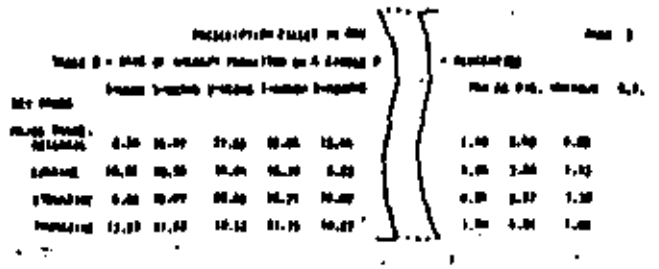


TABLE V—Summary Table (Scientific Application)

BLOOD DISTRIBUTION STUDY - % BREAKDOWN PAGE 1

(REGION) (INPUT) WHOLE BLOOD	JACKSON-VILLE	TUCSON	KANSAS CITY	DALLAS	BALTI-MORE	MEMPHIS	PHILA-DELPHIA	TOTAL
COLLECTED-FACILITY:	63.	0.	0.	162.	532.	624.	1322.	6549.
(% WHOLE BLOOD)	4.74	0.0	0.0	5.18	21.71	47.38	13.83	18.49
RECEIVED:								
WITHIN REGION	1208.	731.	2425.	2510.	1772.	403.	8060.	26592.
(% RECEIV-WB)	95.34	84.61	96.69	84.57	92.34	58.15	97.89	92.11
(% WHOLE BLOOD)	90.83	84.61	96.69	80.19	72.30	30.60	84.34	75.08
(% RECEIV-WB+RC)	57.52	40.59	57.25	66.74	65.36	45.38	86.23	67.98
(% TOTAL INPUT)	49.77	37.85	57.01	59.49	42.79	23.21	66.55	53.05
OUTSIDE REGION	59.	133.	83.	458.	147.	290.	174.	2278.
(% RECEIV-WB)	4.66	15.39	3.31	15.43	7.66	41.85	2.11	7.89
(% WHOLE BLOOD)	4.44	15.39	3.31	14.63	6.00	22.02	1.82	6.43
(% RECEIV-WB+RC)	2.81	7.38	1.96	12.18	5.42	32.66	1.86	5.82
(% TOTAL INPUT)	2.43	6.80	1.95	10.86	3.55	16.71	1.44	4.54
TOTAL RECEIVED	1267.	864.	2508.	2968.	1919.	693.	8234.	28870.
(% WHOLE BLOOD)	95.26	100.00	100.00	94.82	78.29	52.62	86.17	81.51
(% RECEIV-WB+RC)	60.33	47.97	59.21	78.92	70.79	78.04	88.09	73.80
(% TOTAL INPUT)	52.20	44.15	58.96	70.35	46.34	39.92	67.99	57.59
TOTAL WHOLE BLOOD	1830.	864.	2508.	3130.	2451.	1317.	9556.	35419.
(% TOTAL INPUT)	54.80	44.15	58.96	74.19	59.19	75.86	78.90	70.66

TABLE VI—Tables of Contents

GEORGE WASHINGTON UNIVERSITY - - BIOMETRIC LAB

TABLE OF CONTENTS

STUDY: INVESTIGATOR: TITLE:

which could enhance the documentation need of the drug studies data we process, i.e., the implementation of a book system. The Book System searches the completed output files for a given study and then organizes the output in book form with a table of contents, organized and paginated output, and a key terminology index.

EXAMPLE OUTPUT

When the output requires formulation of symmetrical tables, a page image on the printer paper can be envisioned as a field of 132×66 matrix, and any number of sub-fields can be constructed within the field. Table I shows the upper left quadrant of a Pretreatment/Posttreatment cross-tabulations page image. Since the output requires juxtaposition of four identical tables, only this table is generated and then reflected onto three other quadrants for the complete page image. Three types of label (H, S, and O) are identified in this table.

Table II shows calculated results from OOSTAB being merged with the table produced above. One type of label (A) and all three classes of results are identified in this table.

TABLE VII—Subject Age Table (page 10)

STUDY: INVESTIGATOR: TITLE:

APDI - ADULT PERSONAL DATA INVENTORY (FORM 45) FREQUENCY DISTRIBUTIONS

GROUP 1

TABLE NO. 1 ----------SUBJECT AGE

CATEGORY NAME	CODE NO.	FREQUENCY	PER CENT	CUM FREQ.	CUM %	HISTOGRAM OF PERCENT
21 YRS.OLD.	21	5	5.2	5	5.2	:11
23 YRS.OLD.	23	8	8.2	13	13.4	:1111
24 YRS.OLD.	24	11	11.3	24	24.7	:111111
25 YRS.OLD.	25	14	14.4	38	39.1	:1111111
27 YRS.OLD.	27	20	20.6	58	59.7	:1111111111
29 YRS.OLD.	29	16	16.5	74	76.2	:11111111
33 YRS.OLD.	33	13	13.4	87	89.6	:111111
43 YRS.OLD.	43	7	7.2	94	96.8	:11111
44 YRS.OLD.	44	3	3.1	97	99.9	:1

HISTOGRAM OF PERCENT: 10 20 30 40 50 60 70 80

N = 97 MEAN = 28.54 S.D.= 8.00

MISSING DATA CODE= 99 FREQ.= 0 PER CENT = 0.0 TOTAL N = 97

(10) ←

TABLE VIII—Index

STUDY: INVESTIGATOR: TITLE:

I N D E X

Tables III and IV illustrate the tailoring of output according to user specifications. For this application, COSTAB accepted a two-way cross-tabulation of Drug by Duration (42×14) design, results generated and stored in design order; i.e., 14 elements (frequency, percent, cumulative percent, etc.) are stored together for each cell of the design. However, the user wanted the data displayed as separate tables (frequency table, percent table, cumulative percent table, etc.). Here, only one page image was created and results were distributed selectively to the appropriate tables for output.

Table V shows summaries that are frequently used in both industry and science, i.e., percentages, subtotals, and totals.

Tables VI, VII, and VIII are illustrated to show some extracted materials from one of our completely documented studies. (Page 10, ←arrow marked, has been selected to demonstrate page reference capability of this book system).

CONCLUSION

Based on our experience with the GO System, we firmly believe that our approach is a viable one. We continue to develop more general algorithms for the interface executive monitor, so that it will accommodate more flexible output. Descriptors and results files are generated separately and are maintained by trained technicians. Our first generalized computational module (COSTAB) proves that other modules can be readily incorporated into the GO System.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

DESIGN AND IMPLEMENTATION OF AN INFORMATION BASE FOR DECISION MAKERS

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

13

Design and implementation of an information base for decision makers*

by R. H. BONCZEK, C. W. HOLSAPPLE and A. B. WHINSTON

Purdue University
West Lafayette, Indiana

ABSTRACT

When considering the design and implementation of systems for decision support, a crucial point is the power and flexibility of available tools for representing data contexts. The value of such systems is constrained by the "richness" of patterning allowed by their data structure mechanisms. We introduce the notion of an information base as a natural step forward in the continuing evolution of data structures. The outstanding features of the information base are (1) its accommodation of the horizontal and vertical integration of information parcels into a single semantic mechanism, and (2) the integration of operators into this semantic structure.

INTRODUCTION

A topical and decidedly significant area of research involves the identification of those criteria which a computerized information system must satisfy if it is to be of value to non-programming decision makers. The ensuing discussion focuses upon such criteria and their implications for system design and implementation. In particular, we introduce the notion of an *information base* and demonstrate how it may be developed and implemented as an extension to the CODASYL DBTG[1] approach to data management. We commence with the characterization of an information base as a semantic network. It is then shown that this semantic network may be realized as an extension to an approach that has been used in commercial environments. Moreover, we illustrate how the information base serves as the cornerstone for a generalized decision support system.

Within the scope of this paper a distinction is drawn between the terms information and data. Observe, first of all, that information is an abstraction; it is not something which can be pointed to or seen. However it may be conveyed by patterns of "matter-energy,"[2] i.e., by configurations of symbols, by data. Data and information invariably accompany one another. The words on this piece of paper are not information, but rather a pattern of matter-energy which as a consequence of certain activities (e.g., inputing, transmitting, decoding, associating, storing, deciding, etc.) conveys information.[3] The important point is the patterning of data; the "richness" of a notation in terms of the kinds of data relationships which it can represent has obvious implications for its power in conveying information. With this in mind, we can note a pronounced trend in the history of information systems from the relatively impoverished linear data structure to the tree and network data structures, capable of a greater variety of data configuration; correspondingly the ease with which comparatively complex information can be conveyed has also grown. Summarizing, "... we can say that data is an objective notation which has no significance in itself, versus information as a subjective concept which relates a datum to a context."[4]

In order to understand the varieties of contexts or configurations in which data must appear if there is to be a comprehensive conveyance of information, we examine the field of semantics. Of special interest is the notion of a semantic net. The results of this examination constitute a basis for the specification of *information base* features which permit the unambiguous representation of all types of information pertinent to decision support applications. This representation must configure data such that all significant relationships among parcels of information (e.g., among facts, procedures, empirical information, etc.) are accommodated. Furthermore, these objectives for information base features must be met in a manner that is amenable to processing for the purposes of inference and deduction.

Since semantics deals with the *relationships* between symbols and what they denote or mean,[5] what we call the information base may be viewed as a semantic mechanism capable of representing meanings in terms of data configurations. Its storage technique must be general enough to handle the basic kinds of information involved in decision making regardless of the specific decision application. These types of information are: directive information, conceptual information, empirical information, stimulatory information, information about expectations, information concerning valuations, and procedural information. In addition the information base must be flexible enough to represent the often intricate interrelationships among information parcels, relating them so as to capture their full meaning

* Research supported by Office of Water Research and Technology Grant No. 6538-62-1310.

and impact with respect to other parcels of information. This latter point is particularly significant in that it furnishes a basis for the synthesis of separate parcels of information that are all related to the same object, concept, observations, etc.

Woods[5] defines a semantic network to be an attempt to combine into a single mechanism both the ability to store factual knowledge and the ability to model associative connections which render certain parcels of information accessible from certain others. Moreover he indicates three criteria which must be satisfied by a notation used for semantic representation:

1. Logical adequacy. The notation must provide an exact, formal and unambiguous representation of any particular interpretation that may be given to a sentence.
2. There must be an algorithm for translating an initial sentence into this notation.
3. There must be algorithms capable of using the semantic representation in order to perform needed inferences and deductions.

The information base detailed in the subsequent discussion will be shown to satisfy the definition of a semantic network. A query language will be described which, in conjunction with the information base, will be shown to satisfy the three requirements of notations for semantic representation.

FEATURES OF THE INFORMATION BASE

A specific design and implementation of the information base is described in a later section; this design and implementation is based in part upon the idea of a network data base advanced in the CODASYL DBTG Report of 1971[4] and subject to extensions and modifications outlined in Reference 6. The term information base, rather than data base, is used to emphasize its incorporation of two fundamental features which do not appear in the general data base management literature. Both features concern ways of patterning data that can convey information not commonly treated in the guise of data base management, but of value to decision makers; they furnish methods for introducing two novel kinds of context into data structures.

In observing the progression from linear structures to trees, to networks, we note increased facility for relating a datum with other data; there is an increased capacity for specifying the context of a datum in terms of data structures. Though there is little context inherent in linear data structures, the data content of groups of such structures may be used to represent trees. This becomes complex and cumbersome as the tree to be represented grows in size. Similarly, though it is possible to twist tree structures to the task of representing large or complex networks by using collections of tree-like structures, this cannot be accomplished in a facile, straightforward manner. This analogy may be continued with respect to the two features being introduced in this section. That is, they can, in some sense, be represented within network data structures, but such an approach leads to certain asymmetries (with respect to processing) and difficulties akin to those encountered when representing trees in linear structures. Since the two features are not inherent in the common notion of a network data base, we introduce the information base as a mechanism which encompasses both while allowing full network capabilities.

The first feature involves the introduction of the concept of resolution levels within the mechanism for information organization. A simple example of this is described by Winograd.[3] Consider data about cars in which specific weights and colors are related (linked) to each car; on a higher level of resolution, we may want to somehow store information about what the properties of cars are. So on one level of resolution we are interested in specific attributes of specific cars and on another level we are concerned with properties of cars. Thus two distinctive characteristics of the information base are links which integrate individual information parcels on a given level of resolution into a single network structure and secondly, the integration of information of varying levels of resolution into a single structure. We term the former characteristic "horizontal integration" and the latter "vertical integration." So horizontal refers to linkage of entities on the same level; whereas vertical denotes linkage among different levels via information parcels that participate in both levels (though the nature of participation is different on each level). A subsequent section of this paper describes both an implementation and the implications of this feature.

The second outstanding feature of the information base involves its ability to handle the integration of programs into its logical structure. Not only does this permit the linkage of a datum with a program that uses it; it allows the construction of networks (in both the horizontal and vertical sense) of programs. This capacity has two primary effects. First, it provides the basis for model formulation. Second, it furnishes a more comprehensive mechanism for semantic representation.

The aspect of model formulation involves the action of relating certain modules into a desired configuration. This necessitates a knowledge of which configurations are meaningful and which are not. Such knowledge is stored in the information base's semantic network. This approach has much in common with the notion of structured programming. Programs devised according to the tenets of structured programming[7] are readily amenable to storage within the information base; indeed there is also the ability to store alternative modules (e.g., alternative functional forms) for performing a particular role within the context of either other modules or a higher resolution level. The advantages of structured programming in terms of maintainability and extensibility[8] are also apparent in the strategy of integrating program modules into the logical structure of an information base. That is, it is possible to add, replace or delete a module in the same manner that one would add, replace or delete an occurrence of data.

It is useful at this juncture to point out a distinction

between program modularity and program resolution. The idea of resolution level also goes under the name of level of abstraction. Dijkstra[9] indicates that each level of a system's software hierarchy constitutes an abstract resource which participates in the next higher level and which has available to it the resources of lower levels. So "... at one level the programming amounts to manipulation of the abstract resources supported by the next lower level of the hierarchy. The programs at that level manipulate abstractions—the abstractions of the resource, whatever it may be—and at the same time participate in generating a higher level of abstraction for the next level of the hierarchy to manipulate."[10] Furthermore, Miller and Lindamood suggest that a "... highly modular implementation is one in which specific functions are performed by specific modules (and nowhere else); on the other hand, a system which preserves a hierarchy of abstract resources would appear to require modularity as a minimum, and perhaps a great deal more 'structure'."[10] Such a structure is effectively treated by the information base feature of resolution levels which allows the arrangement of program resources into levels of abstraction.

The second effect of allowing the integration of programs into the structure of the information base is the more comprehensive semantic representation that is permitted. Much literature about semantic networks is concerned with the network representation of English sentences (e.g., see References 5 and 11). These sentences consist of patterns of verbs and arguments. The typical decision maker who queries the information base requests the execution of some model (i.e., operators, verbs) using certain data (i.e., operands, arguments) as inputs. The usual data base structures handle information about arguments only; the meaningful operator contexts in which such arguments may appear is not represented in standard types of data base structures. A more detailed discussion and practical application of this feature of representing programs in an information base is presented in Reference 12. The remainder of this paper focuses on details and examples of the resolution level feature and on the utility of the information base as a device for semantic representation.

REQUIREMENTS FOR A DECISION SUPPORT SYSTEM

Recall that, in this paper, we are principally concerned with the information base from the standpoint of its contribution to the realization of a general decision support system. Although there are several facets involved in reaching decisions, we investigate three in particular: information access, model formulation, and analysis. The efficacy of a decision support system may be evaluated in terms of its flexibility, facility, scope, timeliness and cost in supporting these three facets.

With respect to information access there must be a mechanism for the systematic, integrated storage of all pertinent information. The information base outlined above provides just such a mechanism, through both horizontal and vertical integration and through its capacity to relate operators with each other and with arguments. Given such a storage mechanism there must be a technique for interrogating (and modifying) it that can be used by decision makers who are not computer experts or programmers. The query language for accomplishing this is presented later.

The second facet which must be supported is the activity of model formulation. This facet refers both to models that are subsequently used for purposes of analysis and to models in the sense of plans to be implemented. This is a crucial aspect for resolving unstructured problems and for supporting the exploratory aspects of decision making. In short, the decision support system must have a component for the generation and evaluation of alternatives for achieving a stated goal. As already indicated, the information base contributes to such an end.

The decision support system must also provide for the activity of analysis; i.e., the fitting of data with models and models with data, thereby resulting in some expectation, beliefs or knowledge. Implicit in the very nature of the planning activity is the dynamic quality of the interface between model and data; for even though a collection of data may be comparatively stable over some time period, both the problems and the models used for problem solving may be subject to frequent alteration. Notice that a model operates on a particular subset of the entire collection of operands available, and it requires a certain configuration of this data as input. We contend that the tedious, cumbersome task of interfacing data and models for purposes of analysis should be automatically handled by the decision support system in response to the commands of a non-programming user. The method for accomplishing this is discussed in subsequent sections.

FORMALIZATION OF THE INFORMATION BASE

We now present a formal description of what is meant by the term "information base." We define a record occurrence to be a uniquely labeled aggregate of data (i.e., string of symbols). Where I^+ is the set of positive integers, let X_k be the set of labels associated with a finite set of record occurrences, such that $X_k \subset I^+$. A record type, uniquely denoted by the label p_k, may be described by a function r_k as follows. Define R_k as the set of all $r_k: X_k \rightarrow \{0,1\}$ such that:

(1) $\forall x \in X_k;\ r_k(x) = \begin{cases} 1 & \text{if } x \text{ is of the type labeled } p_k \\ 0 & \text{otherwise} \end{cases}$

(2) $\forall x \in X_k,\ \sum_i r_i(x) \leq 1$

(3) $\forall r_i \in R_k,\ \sum_j r_i(x_j) > 0 \quad$ where $X_k = \{x_j\}$

Property (1) states that r_k defines the collection of $x \in X_k$ of the type labeled p_k. Property (2) indicates that each $x \in X_k$ can belong to at most one p_i. Property (3) states that each r_i is non-trivial.

Before defining X_k for $k > 0$, we note that $P_0 = \{p_i\}$ is the set of all labels associated with the elements of R_0. Since X_0

is finite, we can define these labels such that $P_k \subset I^+$, $P_k \cap X_0 = \emptyset$; furthermore we can define each of these sets of labels such that it has no elements in common with any other P_k. Define:

$$X_1 = \{p_1 \in P_0 \mid \exists x \in X_0 : r_1(x) \neq 0\} \cup X_0$$

$$X_2 = \{p_2 \in P_1 \mid \exists x \in X_1 : r_2(x) \neq 0\} \cup X_1$$

$$\vdots$$

$$X_k = \{p_k \in P_{k-1} \mid \exists x \in X_{k-1} : r_k(x) \neq 0\} \cup X_{k-1}$$

It follows from the definition of X_0 and R that there must exist a K such that $X_k = X_{k+1} = \ldots$; then let $X = X_k$. Observe that X is the set of labels of all record occurrences within an information base; these labels are unique identifiers, thereby serving as information base keys. All occurrences of a record type denoted by the label p can be determined by successive applications of the function r to the set X. The magnitude of K indicates the levels of resolution inherent in the information base. The reader will notice that P is always a subset of X; if it were not desired to treat all record types as record occurrences, one could define $X = X_{k-1}$. There are advantages to defining $X = X_k$, especially for purposes of altering the logical structure of an information base after it has been loaded. This will be elaborated in a subsequent section.

Continuing, we now formally define the information-set (In-set). This construct, as implemented in the information base, is drawn in part[6] from the "set" idea of the CODASYL DBTG Report,[1] hence the term "In-set." It is important to differentiate this from the familiar notion of a mathematical set. Let $Q_i = \{x \in X \mid r_i(x) \neq 0\}$. If a function associates each element of its domain with no more than one element of its range it is said to be a functional relation. Then each functional relation $f: Q_j \rightarrow Q_i$ uniquely defines an in-set of which the record type r_i is said to be the owner and the record type r_j is called the member. It is important to make several observations about the in-sets of an information base. It is permissible, and sometimes useful,[6] to allow $i = j$. Second, an in-set may be used to associate record types of different levels of resolution. Third, the set F of in-sets of an information base must be carefully defined so that its elements are consistent; e.g., one should exercise caution in defining both $f_1: Q_i \rightarrow Q_j$ and $f_2: Q_i \rightarrow Q_j$ as elements of F. Finally if $f_1: Q_j \rightarrow Q_i$ and $f_2: Q_i \rightarrow Q_k$, then we can form the composite in-set $f_1 \cdot f_2: Q_j \rightarrow Q_k$ defined by

$$(f_1 \cdot f_2)(x) = f_1(f_2(x)) \; \forall x \in Q_j.$$

This is sometimes desirable from the standpoint of access efficiency; it also allows us to attach special significance or meaning to certain groups of sets.

The foregoing is a formal description of the major features of the information base. It accounts for both the horizontal integration (via in-sets) and vertical integration (via resolution levels) of information into a single mechanism. In order to illustrate the use of resolution levels, we apply the above formalisms to the problem (see Winograd[7]) of representing information about cars. In this problem cars are to be described in terms of color and weight; in addition we would like to denote that color and weight are properties. Suppose we have record occurrences as shown in Figure 1a; these are identified by the respective labels in X_0. The set R_0 is also shown; by inspection we see that R_0 satisfies the needed conditions as given at the beginning of this section. The function r_1 determines whether or not an element of X_0 is of the type color. Similarly r_2 is associated with the type weight and r_3 is associated with the type car. In our implementation each r_i defines (and is defined by) a linked list of occurrences of its type. Given X_0, R_0, and P_0 we apply the rule for defining X_1 to obtain the result shown in Figure 1b. R_1 is also given and clearly satisfies the necessary conditions for its definition. Application of r_4 to elements of X_1 can be used to determine which elements are vehicle properties. Figure 1c gives the X_2 that follows from the definition. If we take $R_2 = \emptyset$, then $X = X_2$.

The occurrences and their "vertical" relations with each other are diagrammed in Figure 2. Also depicted are two in-sets: f_1 and f_2. Using the definitions of Q_1, Q_2, and Q_3 given in Figure 2, $f_1: Q_3 \rightarrow Q_1$ and $f_2: Q_3 \rightarrow Q_2$. The arrows in the diagram point from the owner of the in-set to the member; i.e., each arrow points in the direction opposite to that in the notation of its corresponding functional relation. Using the formalisms introduced here it is a simple matter to represent an extended problem including other kinds of vehicles,[14] more properties, subclassifications of properties (e.g., structural, functional, etc.) and even properties of properties.

An information base for water quality management

More detailed discussions of the water quality management problem may be found in References 14 and 15. The objective of the example presented in this section is to demonstrate the applicability of the information base as a

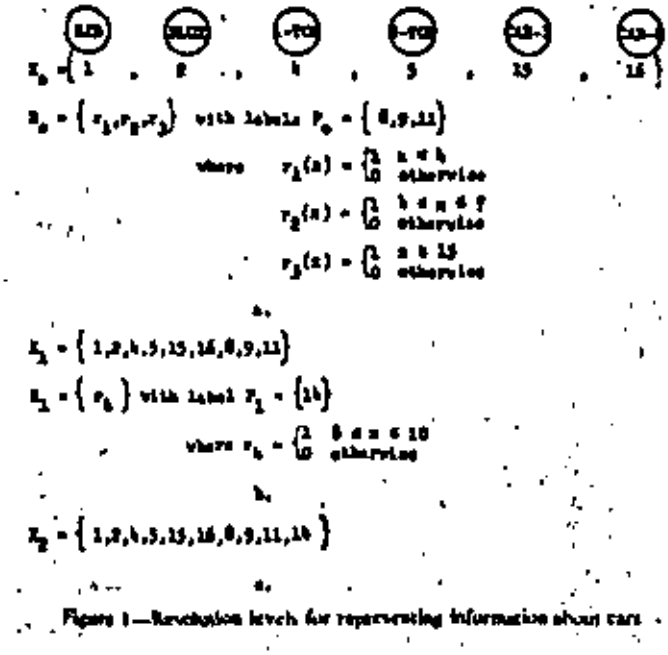


Figure 1—Resolution levels for representing information about cars

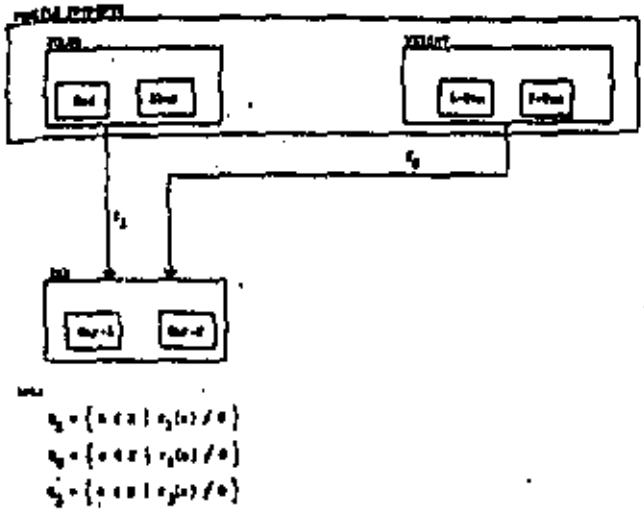


Figure 2—Occurrences in our information base

device for capturing the semantics used to support practical decision problems. At this point, we presume that the reader has a sufficient concept of what an information base entails to obviate the need for complete formalistic description. So for the sake of economy, the following example is presented in a less formal manner than the previous one. It will be used to depict certain implementational details (e.g., Languages in which the information base is specified and with which it is utilized).

Consider the record type POLLUTER, displayed in Figure 3a. This aggregate of data item types represents measures of types of polluter activity for a given date. So occurrences of this record type correspond to measurements taken on various dates. In order to build a semantic network, we must indicate how this concept of POLLUTER fits into the pattern of knowledge concerning water quality management. A polluter is properly characterized as being a property of a river reach. Other properties of a reach include reach parameters, headwater, incremental flow, and treatment plan. So a reach is characterized in terms of these properties as follows: a reach is a portion of a river in which certain water quality parameters are relatively invariant; which has no more than one (point-source) polluter, one incremental flow or one headwater; and which must possess treatment plans. This could be represented in the information base by occurrences of the REACH PROPERTY record type displayed in Figure 3b. However, observe that each occurrence of the data item NAME (e.g., "POLLUTER," "HEADWATER," "PARAMETER," etc.) is also the label of a record type which is itself an aggregate of item types and which may have numerous occurrences. So, for instance, "POLLUTER" denotes an occurrence of REACH PROPERTY; but it also denotes a record type (shown in Figure 3a). The same circumstance holds for the other reach properties, though their record types are not depicted here. The resultant logical structure is illustrated in Figure 3c: a record type enclosed by another record type indicates that the enclosed record type is also an occurrence of the enclosing record type.

We continue the example by examining general water quality modeling characteristics. In order to simulate water quality we need information about the following: the rivers involved, the reaches which are in each river, each reach's properties, junctions, piping plans, and model parameters. This is shown in the structure of Figure 4. Note that the record type GMC has two item types: CHAR (characteristic) and IMPT (a measure of the relative importance of each characteristic). Five occurrences of GMC are shown: RIVER, REACH, JUNCTION, MODEL and PIPE PLAN. General Modeling Characteristic is not the only property of

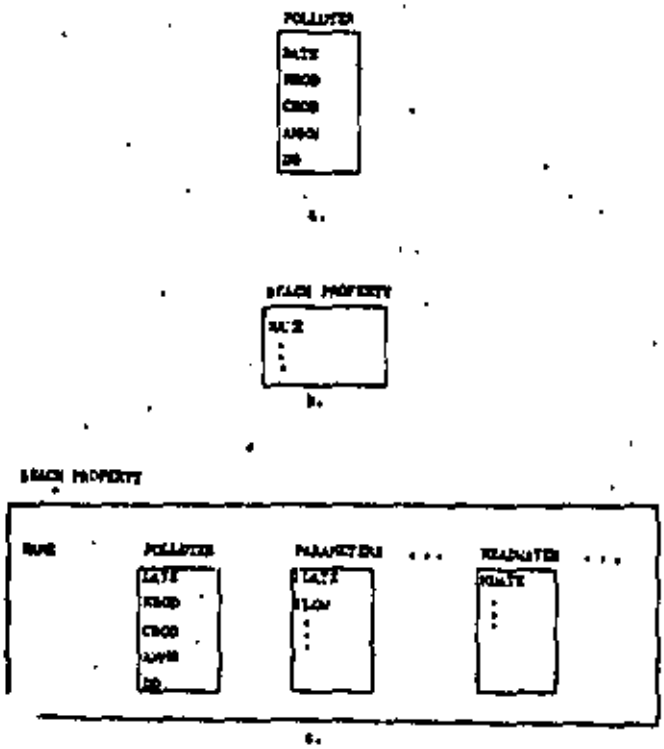


Figure 3—Attributes of a logical structure

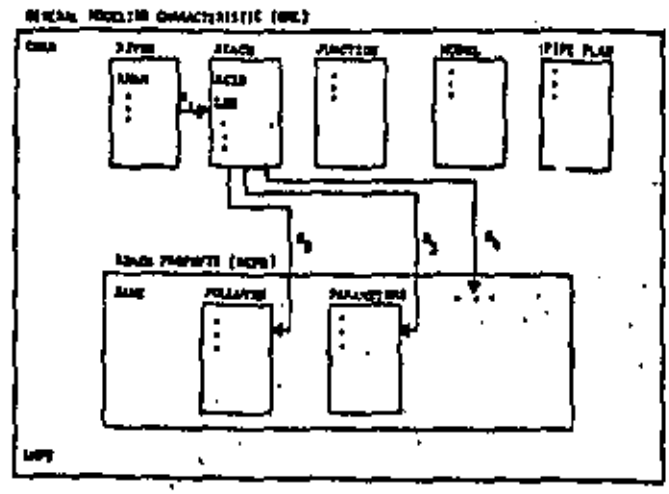


Figure 4—Example of logical structure (GMC)

a segment that needs to be represented; Local Modeling Characteristics (LMC) are also needed. (The term segment is used to indicate a particular area of a river basin.) The details of the high level record type LMC are not shown here, but they describe information about non-point sources of pollution, permits for point-source pollution, treatment plant construction status, permit violation data, etc. As shown in Figure 5, GMC and LMC are occurrences of SEGMENT PROPERTY which is itself an occurrence of the record type WQMA; BASIN and SEGMENT are also occurrences of this record type. The information base could be further extended to incorporate aspects of land use planning since they influence and are influenced by water quality management.

The foregoing logical structures are initially defined in terms of an Information Description Language (IDL). Use of the IDL to define the logical structure of Figure 4 is presented in Figure 6. The specification shown is largely self-explanatory. Each record type is followed by the item types which compose it. If the record type is of a high level, then its item types are followed by a specification of those record types which are its occurrences. Definition of an in-set must be preceded by specifications of its owner and member record types. For simplicity, details of the type and size of items are not shown; also the ordering criterion of each in-set is not shown.

A LANGUAGE FOR DECISION SUPPORT

The reader will observe from the preceding discussion that the decision support system has two basic components: an information base and a query language. Clearly the usability of a semantic network depends upon implementation of a language with which one can extract (insert) meanings that are held in the semantic net. Not only are semantics conveyed by a particular language, they are limited by it as well. The language is used to express meanings, but it also delineates the kinds of meanings which are expressed. We can devise arbitrarily complex semantic networks, but their usability is (from the practical standpoint) constrained by the languages (and language processors) which can be interfaced with them. Observe then that there is a fundamental duality of (1) the language in which ideas are expressed, and (2) the structural representation of ideas in an information base. On the other hand the semantic mechanism must be capable of taking full advantage of the language's power. In the case of the

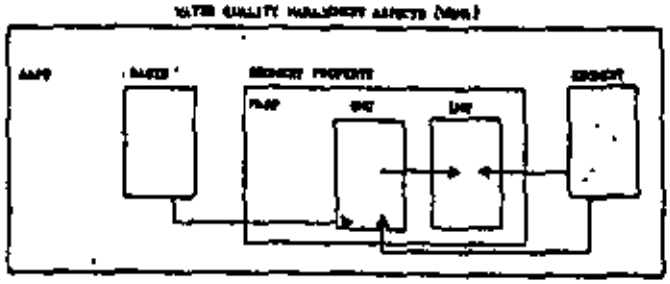


Figure 5—Example of logical structure (WQMA)

```
RECORD      GMC
ITEM        CHAR
ITEM        IMPT
  :
IN          GMC
            RECORD      RIVR
            ITEM        RNAM
              :
            RECORD      RECH
            ITEM        RCHD
            ITEM        LEN
              :
            RECORD      JUNC
            ITEM        JID
              :
            RECORD      MODL
            ITEM        MID
              :
            RECORD      PIPE
            ITEM        PPID
              :
SET         S1
OWNER       RIVR
MEMBER      RECH
RECORD      RCPR
ITEM        NAME
  :
IN          RCPR
            RECORD      PLTR
            ITEM        DATE
              :
            RECORD      PARA
              :
            RECORD      HDW
              :
SET         S2
OWNER       RECH
MEMBER      PLTR
SET         S3
OWNER       RECH
MEMBER      PARA
  :
```

Figure 6—Example of IDL for Figure 4

implementation described in this paper, the query language is the constraining factor since it is intended primarily for the practical support of decision activities of managers in both the public and private sectors.

Implementation of a natural language (e.g., English) processor is certainly a noble objective. It is our experience that the typical decision maker neither uses, nor needs, a complete facility for conversing in a natural language. It often happens that phrases or clauses are sufficient to convey an idea; there are grammatical constructs (e.g., reflexive, passive) which are not particularly germane to the decision activities of information access, model formulation, and analysis. In addition the decision maker is more prone to desire information conveyed in a tabular or graphical fashion than in a narrative mode. It has also been found that the user sitting at a computer terminal has a tendency to use abbreviations and concise mathematical notation.

With these factors in mind, the query language to be outlined here has been designed to meet the needs of decision makers for flexibility and brevity of expression, while at the same time being easy to learn and utilize. The query language is effectively a subset of English that has been extended to include standard mathematical operators (i.e., relational, arithmetic, and univariate and multivariate functions). The focus here is upon use of this language for interrogation, though it may be used for data creation and modification as well.[9]

The standard framework of the language consists of a collection of operators (used in the capacity of verbs, adverbs and adjectives) relating to operations typically performed by most decision makers; these operators are of two kinds: commands (e.g., LIST, PLOT, STAT, REGRESS, etc.) and mathematical operators (e.g., MAXIMUM, AVERAGE, =, +, <, etc.). In addition to this standard framework the user may define arguments (used in the capacity of nouns), synonyms for arguments and operators, and any further operators (i.e., programs to be integrated into the information base) that are mundane to the particular decision making application to be supported. This definition is effected in terms of an Information Description Language (IDL) which establishes the context(s) of all data, arguments and operators. That is, it defines the semantic net.

Details of the query processor are not discussed in this paper but may be found in References 16–18. Briefly, the query language has a context-sensitive grammar; inverse transformations are used to take a surface structure query into a deep structure expression in a language having a context-free grammar. This deep structure expression is compiled using well-known methods of syntax-directed analysis. Parts of the compiled expression are used as input to network traversal routines which make extractions from the information base for use in analysis indicated by the query's command (verb).

The query's syntax appears as follows:

(COMMAND)(FIND clause)(CONDITIONAL clause)

or alternatively,

(CONDITIONAL CLAUSE)(COMMAND)(FIND CLAUSE).

So some sample queries are:

LIST REACH.NAME,REAERATION.PARAMETER AND REAERATION.EXPONENT. FOR DATE=110175 AND REACH.LENGTH<.9

WHEN DATE=110175, PLOT REACH.NUMBER VERSUS AMMONIA.CONCENTRATION AND DO.CONCENTRATION/LOG(TEMPERATURE)

LIST GENERAL.MODELING.CHARACTERISTIC IF IMPORTANCE>3

The language allows any meaningful configuration of arguments and mathematical operators to appear in the FIND and CONDITIONAL clauses.

PROCESSING HIGHER LEVEL RECORD TYPES

Upon receipt of a query, the query processor generates appropriate commands for traversal of a multi-level network. These commands are operators in an Information Manipulation Language (IML). We use the term IML to distinguish from the Data Manipulation Language (DML) proposed in the CODASYL DBTG Report.[3] The DML is intended to permit access, modification and retrieval for a single level network data base. The IML has the more extensive function of furnishing tools for manipulation of the information base. Thus the IML contains operators for handling traditional DML functions[10] and operators for processing higher level record types. The latter are discussed here.

In the DML, routines exist for creating a record occurrence at a unique location denoted by its key. The IML includes analogous routines for specifying that an existing record occurrence be treated as a record type as well. There are four such commands:

CRTK—Create Record Type based on a given Key

CRTR—Create Record Type based on the current occurrence of another Record type

CRTO—Create Record Type based on the current Owner of a given set

CRTM—Create Record Type based on the current Member of a given set

Another traditional DML operator (AMS) adds a specified record occurrence as a Member of a given Set. A similar IML operator is used for adding an existing occurrence of one record type as an occurrence of another record type. Note that utilization of this operator must be preceded by a generalization of the definition of a record type which was introduced above (i.e., the definition is generalized by removing the restriction that $\Sigma_i r_i(x) \leq 1$, $\forall x \in X_R$, where $r_i \in R_i$). This operator is AORT, Add Occurrence to Record Type, and it uses the key of the occurrence to be added. In conjunction with commands for the logical restructuring of a network data base,[5] AORT provides the ability to add and delete higher level record types and add existing occurrences to higher level record types; and this is accomplished without dumping and reloading data.

Finally operators are needed for determining the key of a record type, given an occurrence of the record type. These commands are:

GKRR—Get Key of the Record type for the current Record occurrence of that type

GKRO—Get Key of the Record type whose occurrence is the current Owner of some set

GKRM—Get Key of the Record type whose occurrence is the current Member of some set

These operators provide the capacity to proceed from a lower level occurrence to a higher level occurrence, when used in conjunction with traditional DML operators.

It must be emphasized that the typical user of the query system needs to have no knowledge of the IML operators, for they are automatically set up and executed by the query processor in response to a user query.

ADVANTAGES OF THE RESOLUTION LEVEL FACILITY

We contend that the concept of resolution levels effectively adds a new dimension to the field of information storage. The preceding discussion has suggested a means for operationalizing this concept as an extension to the traditional single-level network approach. One advantage is that multi-level semantic networks may be stored without introducing asymmetry in the interpretation and processing of in-sets and record types. Since a record type may also be defined to be an occurrence of a higher level record type, the addition of a record type is treated by creating a new record occurrence at the next higher level. That is, we remove the distinction between data values and the structural pattern according to which data is organized. In other words, the terms "attribute" and "value" are recognized as being relative, so that what is a value on one level is an attribute on another and vice versa.

From one viewpoint this abolishes the special status of an IDL specification by permitting record type definition to be a dynamic process. That is, the creation of a new record type is synonymous with the creation of a new record occurrence of a higher level record type. Thus the IDL specification of the highest level of resolution is effectively reduced to the definition of three record types (one describing information about record types, another relating to information about sets, and one with various system information[6]) and some in-sets between them. This definition is always the same regardless of the content and structure of lower resolution levels.

A second advantage, already mentioned in connection with integration of programs into the information base, concerns a mechanism for handling levels of abstraction in software. A third advantage is that higher level record types may be used to characterize areas of an information base by assigning record types of a particular area to be occurrences of a higher level record type; these areas may be defined for a variety of reasons (e.g., for information security, to denote scenarios, to delimit functional areas—which may overlap, etc.). As the information base becomes large and varied in content, this technique may also be used to realize efficiencies in path determination processing by limiting the scope of network traversal to a particular information base area.

THE INFORMATION BASE AS A DEVICE FOR SEMANTIC REPRESENTATION

With the foregoing background, we can now address the three criteria proposed by Woods,[6] which must be satisfied by a notation used for semantic representation. First observe that the information base is a tool for the representation of a semantic network (i.e., a single mechanism with both the ability to store factual knowledge and the ability to model associative connections which render certain parcels of information accessible from certain others).

The first criterion of a notation for semantic representation is logical adequacy. The notation must provide an exact, formal and unambiguous representation of any particular interpretation that may be given to a sentence. Recall that the sentences with which we are concerned are those allowed in the query language for decision makers.

The information base allows a given query to have a multitude of interpretations. The query specifies a group of data items which may be related to each other in many ways via vertical and horizontal linkages in the information base. Each path of linkages on which these items lie corresponds to a particular interpretation of the query. Upon receiving a query which is subject to multiple interpretations the query processor prompts the system's user in order to ascertain which interpretation (i.e., path) is intended. Details of the manner in which this has been implemented may be found in References 16 and 18.

The second criterion is that there must be an algorithm for translating an initial query into the notation of the information base. This is the central function of the query processor whose operation has already been described; implementational details appear in Reference 13. The third criterion, concerning algorithms capable of using the semantic representation, has also been addressed in the discussion of the query language. Observe that the IML provides the means for interfacing algorithms with the semantic representation.[16] Algorithms which have been used range from relatively commonplace report generators to large scale water quality simulation models.[15]

CONCLUSION

In the beginning we observed that it is the patterning of symbols which can convey information; a datum's meaning derives from its context, from its relationships with other data. Thus when considering the design and implementation of systems for decision support, a crucial point is the power of available tools for representing contexts. The value of such systems is constrained by the "richness" of patterning allowed by their data structure mechanisms. Observing the progression from relatively impoverished linear structures to trees and networks, we note that each stage has provided a more powerful and flexible tool for semantic representation. In this paper we have introduced the notion of an information base as a natural step forward in the continuing evolution of data structures. An outstanding feature of the information base is its accommodation of both the horizontal and vertical integration of information parcels into a single mechanism. An information base implementation which builds upon network concepts was discussed. A topic for future research is the investigation of an information base implementation which builds upon the relational data base notions.[19] A second distinctive feature of the information base, namely the integration of operators into its structure, was briefly described. The information base is utilized by a non-procedural, English-like query language, that has been designed for decision support applications. This language, in conjunction with the information base, satisfies the requirements for a notation for semantic representation.

ACKNOWLEDGMENTS

The authors are indebted to Professor Walter Reitman of the University of Michigan, Professor Tibor Vamos of the Hungarian Academy of Sciences and Dr. Bertram Raphael of SRI for useful discussion on semantic network and data base management.

REFERENCES

1. CODASYL, Data Base Task Group Report, ACM, April 1971.
2. Miller, J., "Living Systems: The Organization," *Behavioral Science*, Vol. 17, January 1972.
3. Kneifel, A. M., "Hard vs. Soft Information," *Management Datamatics*, June 1976.
4. Alberda, J. D., *Structures and Relations in Information*, Groningen, Druk: V.R.B. Offsetdrukkerij, Rotterdam, 1974.
5. Woods, W. A., "What's in a Link: Foundations for Semantic Network," in *Representation and Understanding* (ed. Bobrow, D. G. and A. Collins), Academic Press, New York, 1975.
6. Bonczek, R. H., C. W. Holsapple, and A. B. Whinston, "Extensions and Corrections for the CODASYL Approach to Data Base Management," *International Journal of Information Systems*, Vol. 2, 1976.
7. Winograd, T., *Understanding Natural Language*, Academic Press, New York, 1972, pp. 23–27.
8. Dijkstra, E. W., "Notes on Structured Programming," T.H.E. Report No. EWD-248, 70-WSK-03, 2nd Edition, April 1970.
9. Donaldson, J. R., "Structured Programming," *Datamation*, December, 1973.
10. Miller, E. F. and G. E. Lindamood, "Structured Programming: Top-down Approach," *Datamation*, December 1973.
11. Heidorn, G. E., "Automatic Programming Through Natural Language Dialogue: A Survey," *IBM Journal of Research and Development*, July 1976.
12. Bonczek, R. H., C. W. Holsapple and A. B. Whinston, "Implementation of a Decision Support System for Regional Water Quality Planning," Krannert Institute Paper No. 570, Purdue University, West Lafayette, Ind., September, 1976.
13. Bonczek, R. H., "Theoretical Description of Access Language for a General Decision Support System," Doctoral Dissertation, Purdue University, 1976.
14. Haseman, W. D., C. W. Holsapple and A. B. Whinston, "Implementation of a Large Scale Water Quality Data Management System," *Socio-Economic Planning Sciences*, Vol. 10, March 1976.
15. Holsapple, C. W. and A. B. Whinston, "Decision Support System for Area-wide Water Quality Planning," *Socio-Economic Planning Sciences*, Vol. 10, 1976.
16. Haseman, W. D. and A. B. Whinston, *An Introduction to Data Management*, Richard D. Irwin Co., Homewood, Illinois, 1977.
17. Bonczek, R. H., W. D. Haseman and A. B. Whinston, "Structure of a Query Language for a Network Data Base," Technical Report, Krannert Graduate School of Management, Purdue University, April 1976.
18. Bonczek, R. H., W. D. Haseman and A. B. Whinston, "Automatic Path Determination in a Network Data Base," Technical Report, Krannert Graduate School of Management, Purdue University, April 1976.
19. Codd, E. F., "A Relational Model of Data for Large Shared Data Banks," *Communications ACM* 13 (6), June 1970.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

A MULTI-LEVEL PROCEDURE FOR DESIGN OF FILE ORGANIZATIONS

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

A multi-level procedure for design of file organizations

by EIVIND AURDAL and ARNE SØLVBERG
The University of Trondheim
Trondheim, Norway

ABSTRACT

This paper describes a multi-level procedure for design of file organizations. Necessary description of the application systems is discussed and a design procedure outlined. The main levels of the design procedure are:

- normalization of messages,
- synthesis of a logical model of the file organization,
- making a "best possible" physical realization of the logical model using a particular DBMS.

The final solution is evaluated through a performance analysis.

INTRODUCTION

The design of large data bases involves a wide range of problems, from the application system specification to the choice of hardware. The work reported here is restricted primarily to the problem of designing a "best possible" file organization given the application systems retrieval requirements, and given a particular database management system (DBMS) for the implementation. A multi-level approach to the design of file organizations is proposed.

Using a multi-level approach to the design of file organizations the following advantages can be achieved

- transparency of the design procedure,
- elimination of non-effective solutions at an early stage,
- hardware/software selection can be postponed until the appropriate design level is reached.

The multi-level design procedure that is proposed here consists of

- specification of the information processing problem,
- transformation of the specified information structures into a logical model of the file organization,
- modification of the logical model to fit a particular DBMS,
- physical implementation of the modified model using a particular DBMS,
- evaluation of the final solution using a performance analysis.

The various levels of the design procedure are described with respect to the decisions which have to be made on each level, the necessary specifications of the application problem, and the DBMS specifications.

THE INFORMATION SYSTEM SPECIFICATIONS

We shall concentrate on that part of the information system specifications that are relevant to data base design. The specifications must model that part of the real world which we are interested in, the object system. The object system consists of a finite number of objects. An *object* is a unique part of the real world. It is something we are interested in, something we want information about. Objects may be concrete as well as abstract entities, like persons, enterprises, orders etc. Two kinds of features of an object are of interest: the properties of the objects and the relationships between the objects.

Objects of a system can be partitioned into object classes. An object class defines a set of objects which, for our purpose, are supposed to have so many common properties and relationships that we assign one common type of identifier to each object in the set.

Relationships between objects can be described by four different types of binary relations:

- 1:1 *one-to-one relation* describes a relationship between an object of one class and an object of the same or a different object class.
- 1:n *one-to-many relation* describes a relationship between one object of one object class and many objects of the same or a different object class.
- n:1 *many-to-one relation* is the inverse of the 1:n relation described above.
- n:n *many-to-many relation* exists if both the relation and its inverse are 1:n related.

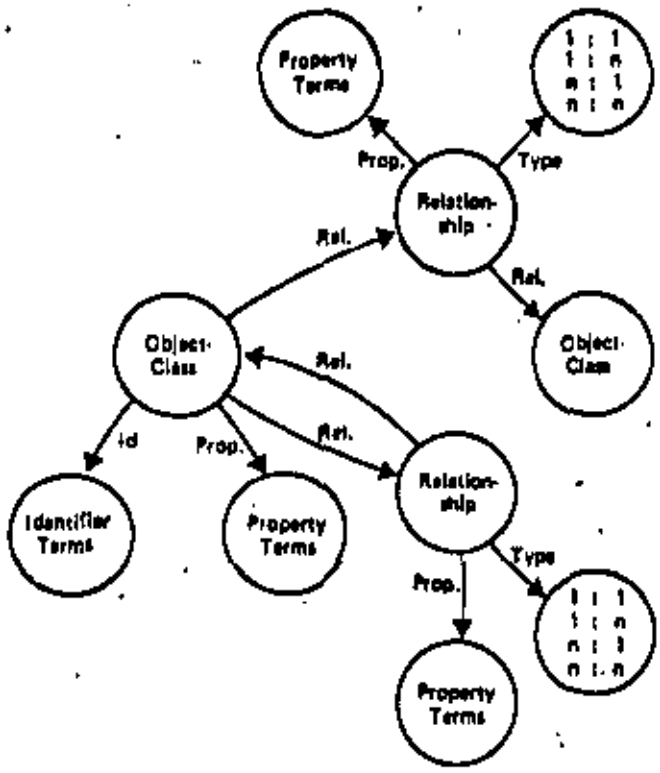


Figure 1—Features of an object

The different features of an object are illustrated in Figure 1. The figure shows how objects are related to other objects, and how identifier-items and property-items are attached to objects. The object description is illustrated with an example in Figure 2. The example shows that a customer may have a number of orders. The customer is identified by "customer no." and described by the property items "name" and "address." One order is identified by "order no" and has the property items "total sum" and "order spec."

In an information processing system, information about real world objects are stored in permanent files in a DBMS. Information about objects is exchanged between different parts of the information processing system by exchanging messages about those objects between processes. A convenient way of specifying this information flow may be

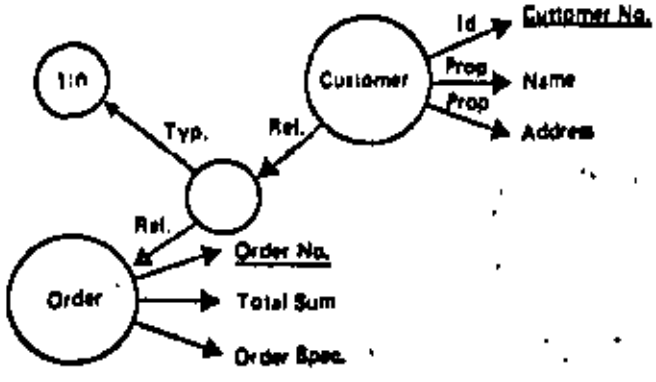


Figure 2

obtained using the principle of hierarchical systems partitioning. This will lead to a level-by-level more detailed specification of the application system, where the terminal elements will be subprocesses and logical files.

We shall illustrate our proposal for a multi-level file-design procedure by a case, which will be a simplified model of a warehouse (Figure 3). The system handles customers' orders for goods, it controls quantity in stock, and it produces refill orders for the vendors.

Suppose that a further detailing of the ORDER MANAGEMENT system results in the subsystem structure of Figure 4. The logical files "STOCK FILE" and "CUSTOMER FILE" represent permanent information, while the logical files "ORDERS," "ORDER ACCEPTED," "ORDER REFUSED" and "ORDER REGISTERED" represent message exchange between processes.

The elements of the logical files can be described by message types.

If we suppose that one customer may give several orders, and that one order may consist of a number of orderlines, the CUSTOMER FILE elements can be specified by the message type:

MT(CUSTOMER FILE):
customer no, customer name, customer address,
{order no, total price,
{order line no, article no, article name,
line price, number ordered}}

where identifiers are underlined and brackets { } represent repeating groups.

Each of the permanent files have to be specified in this way. The permanent files are the basis for the design of the

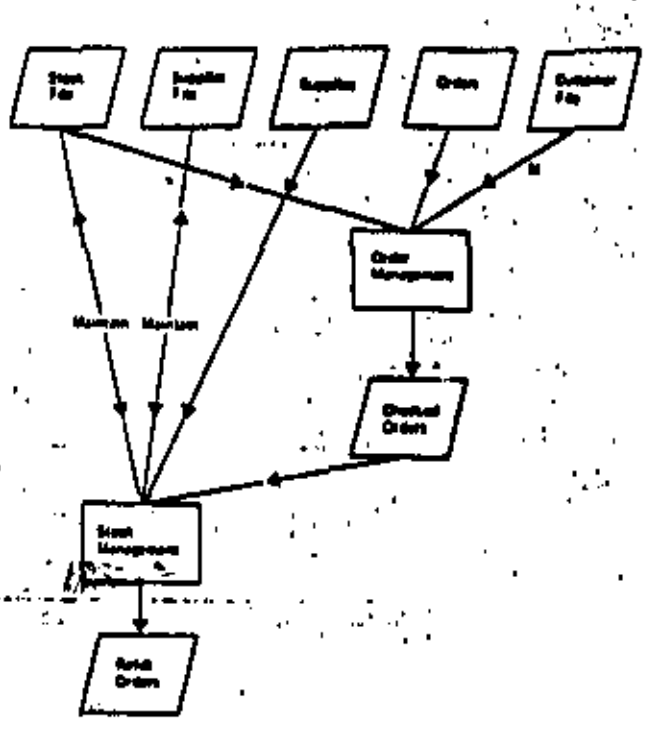


Figure 3

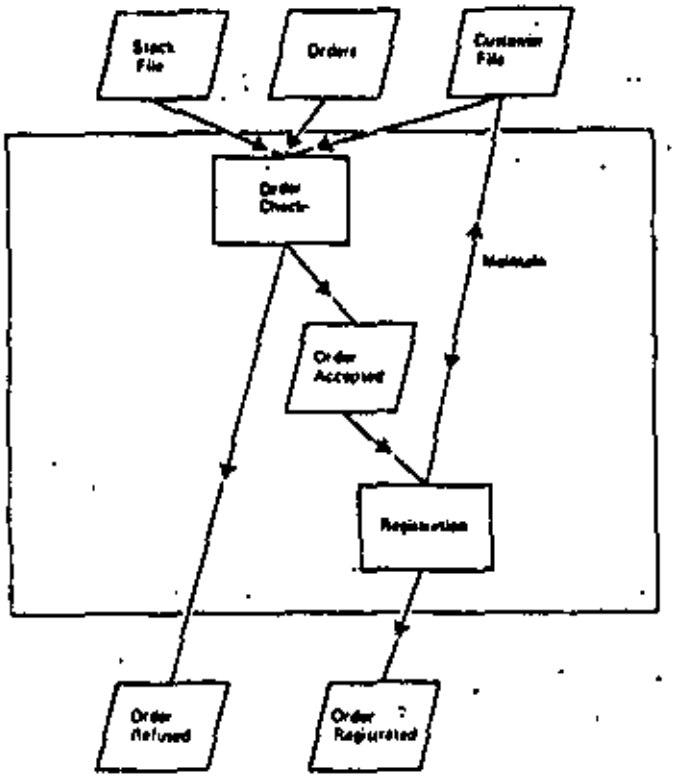


Figure 4—Order-handling system

le organization. Therefore, one has to describe the activity etween them and the information system processes. This can be done with special types of processes, called *retrieval processes*. Each information system process may contain a number of retrieval processes.

Each of the retrieval processes must be one of the following four file operations:

READ,	the retrieval process reads element(s) from a permanent file.
UPDATE,	the retrieval process changes one or more property terms of an element of a permanent file.
WRITE,	the retrieval process stores new element(s) into a permanent file.
DELETE,	the retrieval process removes element(s) from a permanent file.

A retrieval process must have exactly specified search keys. The expected result must also be specified. This specification may be a description of which parts of the permanent file shall be read, or it may be a description of the message(s) which shall be stored.

The system of Figure 4 consists of two subsystems, ORDER CHECK and REGISTRATION. The subsystem ORDER CHECK handles orders from the customers. When an order arrives, the CUSTOMER FILE is checked see if the customer is already registered. If not, some —tion has to be taken, for instance, to refuse the order, or to produce a customer number for this customer. Before the order can be accepted, an order number has to be produced, and the quantities in stock of the specified articles have to be checked. This subsystem will therefore contain two retrieval processes, one for checking the customer file, and one for checking the stock file. The keys are denoted by K (=key) and the expected result by O (=output).

S1: Check customer-file
Operation: READ
K: —customer name
O: —customer no.
S2: Check stock-file
Operation: READ
K: —article no.
O: —quantity

The system REGISTRATION stores new orders and possible new customers. The registration of new orders can be illustrated as follows:

S3: Order registration
Operation: WRITE
K: —customer no.
—order no.
O: —order no.
—total price
—order-line no.
—article no.
—article name
—line price
—number ordered.

(—order-line no. through —number ordered.: for each order-line)

If all the retrieval processes defined by the application system are described in this way, then the application programs can be considered to consist of retrieval processes working towards a central data base containing permanent files (Figure 5). Through the specification of the application problem, we have now developed a basis for the design of a first logical model of the file organization.

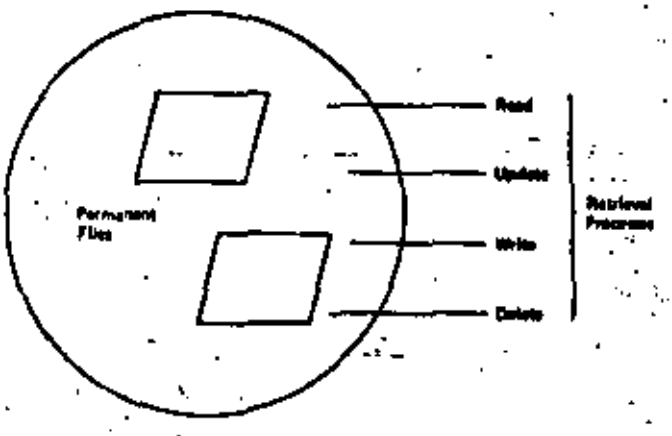


Figure 5—The basis for designing a logical data base model

DESIGNING A LOGICAL DATA BASE MODEL

A logical model of the file organization must satisfy the following requirements:

- It must be entirely based upon the information analysis (i.e., the a priori knowledge of the application problem).
- It must satisfy the requirements specified in the information analysis, i.e., the requirements of information structure and the requirements of retrieval.
- It must involve a "best possible" independence between the designed data structures and the specified application programs.
- It must be easy to realize using any particular DBMS.

The design of the logical model is done in three steps:

(1) Decomposition of the message types
(2) Synthesis of the decomposed messages into a model which satisfies the information requirements and finally
(3) A modification of the model in order to satisfy the retrieval requirements.

The decomposition procedure is similar to the normalization procedure proposed by Codd,[1] which results in an elementary file system.[8] The normalization of message types is a three-step procedure:

1. Elimination of repeating groups, hierarchical structures, or network structures,
2. Elimination of non-full dependence on the primary key,
3. Elimination of transitive dependence between the property terms.

The decomposition will be illustrated with an example. The permanent file CUSTOMER FILE, was described earlier in the following way:

$M1_0$: *customer no*, customer name, customer address, (order no, total price, (orderline no, article no, article name, line price, number ordered))

In this case a repeating group is subordinate to another repeating group, and the first step is to eliminate the first repeating group. The message type is therefore split into the message types:

$M1_1$: *customer no*, customer name, customer address
$M1_2$: *customer no, order no*, total price, (orderline no, article no, article name, line price, number ordered)

The description of an order is only dependent on "order no," not "customer no." The message type $M1_2$ is therefore split into:

$M1_{21}$: *customer no, order no*
$M1_{22}$: *Order no*, total price, (orderline no, article no, article name, line price, number ordered)

An elimination of the last repeating group results in:

$M1_{221}$: *order no*, total price
$M1_{222}$: *order no, orderline no*, article no, article name, line price, number ordered

In message $M1_{222}$ there is a transitive dependence between the property terms, "article name" is dependent on "article no," not the identifier of the message. The message type is therefore split into:

$M1_{2221}$: *order no, orderline no*, article no, line price, number ordered
$M1_{2222}$: *article no*, article name

Using this kind of normalization procedure, the message type $M1_0$ has been decomposed into the five message types $M1_1$, $M1_{21}$, $M1_{221}$, $M1_{2222}$ and $M1_{2221}$. The decomposition was entirely based upon specification of the application systems requirements, i.e., information about objects and message types. Normalization is a reversible process. That means that the original message types can be reconstructed, and therefore no information has been lost.

In the example of Figure 3, the system contains, in addition to the CUSTOMER FILE, also the permanent files STOCK FILE and SUPPLIER FILE. Suppose that the permanent file STOCK FILE contains messages which describe the various articles in stock, and that each article may have several substituting articles and several suppliers. A possible decomposition might be:

$M2_1$: *article no*, article name, quantity, price
$M2_2$: *article no, article no*
$M2_3$: *article no, supplier no*

Suppose the SUPPLIER FILE contains messages which describe the various suppliers and that one supplier may deliver several articles. That supplier may also have several refill orders registered. A possible decomposition might be:

$M3_1$: *supplier no*, supplier name, supplier address
$M3_2$: *supplier no, article no*
$M3_3$: *supplier no, refill order no*
$M3_4$: *refill order no*, number wanted, article no

Some of the message types are binary relations which represent relationships between objects. For example: $M1_{21}$: *customer no, order no*, represents a binary relation between the objects of the object classes CUSTOMER and ORDER. Other messages may describe properties of an object, for example $M1_1$: *customer no*, customer name, customer address.

Examining the message types, one may determine the

relationships between the various messages. This will be illustrated with examples:

(1) The message type
$M1_{n:1}$: *customer no, order no,*
describes an 1:n relationship between messages identified by "customer no" and messages identified by "order no."

(2) In the message type
M3: *refill order no*, number wanted, article no.
the secondary key, "article no," is used as a property term. This indicates an n:1 relationship between messages defined by "refill order no" and "article no."

The result of such an examination is illustrated in Figure 6. One may observe that there are both a 1:n relationship and a n:1 relationship between "article no" and "supplier no." This implies the existence of a n:n relationship.

The logical model may be represented in a convenient way, using a data structure diagram. The data structure diagram has two basic elements: boxes and arrows. Each box represents a class of records (or a file) and each arrow represents a meaningful relationship between records. Such a diagram may easily be drawn using the normalized message types. Message types may be represented as boxes, and a 1:n relationship is an arrow between boxes. A n:n relationship is represented as a coupling (or a relational file) between boxes.

A data structure diagram based upon the normalized message types of the permanent files CUSTOMER FILE, STOCK FILE and SUPPLIER FILE is shown in Figure 7. This is a logical model which satisfies the requirements of the information structure.

	Customer No.	Order No.	Order No. Orderline No.	Article No.	Refill Order No.	Supplier No.
Customer No.		1:n				
Order No.			1:n			
Order No. Orderline No.				n:1		
Article No.				1:n		1:n
Refill Order No.				n:1		
Supplier No.				1:n	1:n	

Figure 6—Relationships between identifiers of the normalized messages

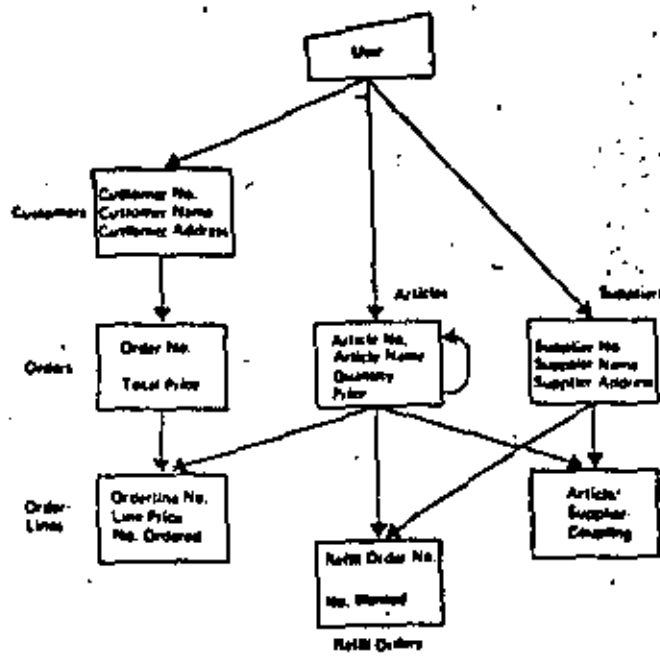


Figure 7—Data structure diagram of the normalized information structure

The next step of the design procedure is aimed at satisfying the retrieval requirements. Suppose that both "customer no" and "customer name" are used as keys by different retrieval processes, and that the response time requirements exclude a sequential processing of the CUSTOMER FILE. A modification of the data structure diagram is then necessary (Figure 8).

A search file consisting of "customer name" is established and 1:n-related to the customer file. Usually, the retrieval requirements may be satisfied establishing search files in a way similar to the one illustrated in Figure 8.

Suppose further examinations of the retrieval processes show that the items "article no.," "article name," "supplier no" and "supplier name" are retrieval keys. Necessary modifications of the logical model may result in a data structure diagram as shown in Figure 9.

A logical model has now been developed in three design steps. The first step, decomposition of messages, was based on message description and object description. The synthesis of the decomposed messages was based upon relation-

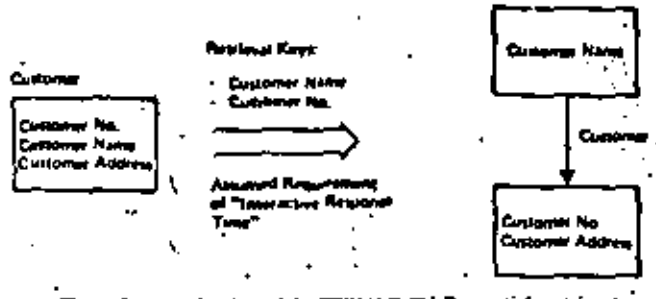


Figure 8—Modification of CUSTOMER FILE to satisfy retrieval requirements

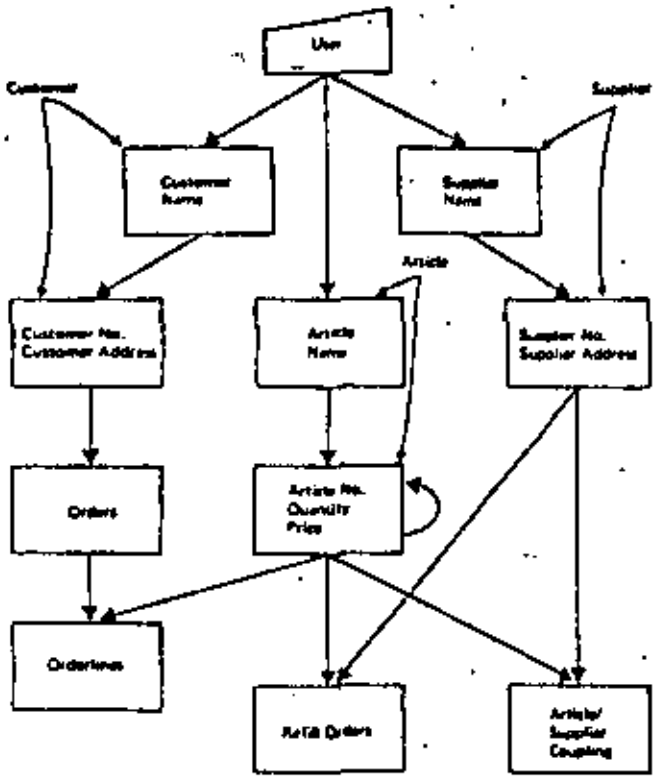


Figure 9—Logical model which satisfies the retrieval requirements

ships between objects, and the resulting model was modified, using information about the retrieval processes and the response time requirements. Thus, a logical model of the file organization has been developed, without making any decisions about any particular DBMS. The entire design process is reversible. Thus, the original information structure can be reconstructed, and the requirements stated in the information analysis is therefore satisfied.

FITTING OF THE LOGICAL MODEL TO A PHYSICAL MODEL

The next step in the design procedure is to select a particular DBMS, and modify the logical model in order to give a "best possible" physical solution. In this paper the efforts will primarily be devoted to DBTG-like DBMS-software products, but most of the problems will nevertheless be of a general character.

The available DBMS may have certain restrictions, like:

- the DBMS does not allow network-structures, only tree-structures,
- the DBMS allows direct access to only a limited number of hierarchical levels,
- the DBMS allows only a certain number of hierarchical levels
- combinations of the restrictions mentioned above.

Therefore, the first modification of the logical model is to satisfy the DBMS restrictions. Substitutions of lists by inverted lists will usually solve these problems.

The next step is to make certain adjustments of the logical model in order to:

- optimize the number of block accesses,
- optimize the data volume,
- optimize the transport of data between memory and the data files.

Unfortunately, these factors are conflicting, and an operation which takes all the factors into consideration must be carried out. One may also notice that the less complex a file organization is, the more it simplifies such operations as initial loading, reorganization and report generating.

The logical model describes the record design and the choice of access paths. Any adjustment of the logical model must therefore either modify the record design or the choice of access paths. A modification of the record design which has to satisfy the information structure requirements must be a choice between storing of duplicate data item values or the use of references. The modification of the access paths is a choice between the use of lists or inverted lists. Duplicate storing of data will be illustrated as shown in Figure 10.

The present version of the logical model describes the customers as illustrated in Figure 10(a). Suppose many retrieval processes are frequently using "customer no" as key and want information about "customer name." Such a situation will result in a large number of accesses from the member records to the owner records. An alternative version of the record layout is illustrated in Figure 10(b). The item "customer name" is stored in both the owner records and the member records. Such a solution will result in a decrease in the number of accesses, an increase in the file size, and an increase in the data read and written. In order to find the best solution one has to estimate the decrease in access costs versus the increase of input/output and storage costs.

Consequently, at this level of the design procedure, there is a need for an analysis tool which may be used in making such estimates. The analysis tool must satisfy the following

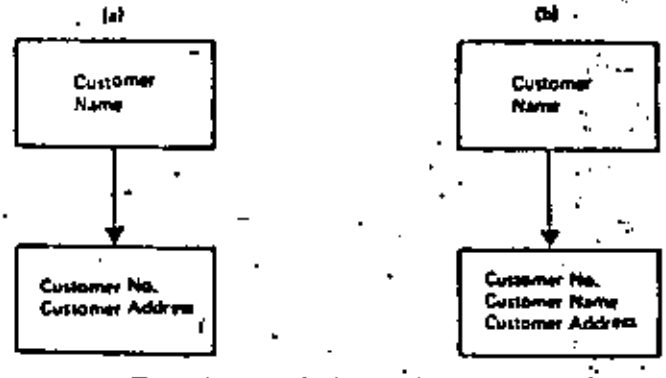


Figure 10—Alternative layouts of the customer file (

conditions:

- It must be possible to perform an analysis without making any decisions about physical realization.
- the analysis must be based on a description of the logical model and the retrieval processes.
- the analysis must form a basis for making decisions of the type mentioned above.

The logical model describes the file organization, and the retrieval processes describe the activity against the file organization. Consequently, it is possible to describe the activity between the different record types and the activity between records within the same record type. It will be shown later that such a description is useful in making decisions about logical restructuring or about physical realization.

A part of the order management system described in the previous sections is shown in Figure 11. Suppose the following retrieval process exists:

S1: Read order
Operation: READ
K: —order no
O: —order no
—total price
—orderline no
—article no
—article name
—line price
} for each orderline

The process uses "order no." as key, and its access path is illustrated with a dotted line in Figure 11. One may observe that the process results in transitions between the record

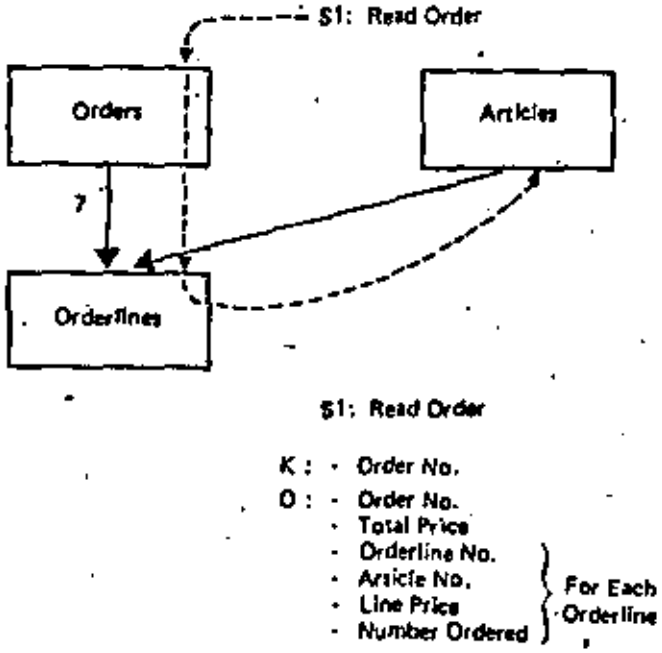


Figure 11—Illustration of the access path of a retrieval process

types ORDERS and ORDERLINES and between ORDERLINES and ARTICLES. These transitions are called retrievals of the first kind. An order will usually consist of more than one orderline, and there will be transitions between the records in the record type ORDERLINES. These transitions are called retrievals of the second kind. Suppose the average number of orderlines per order is seven. The retrieval process S1 then will result in one retrieval of the first kind between user and ORDERS, the same between ORDERS and ORDERLINES, six retrievals of the second kind within the record class ORDERLINES, and finally seven retrievals of the first kind between ORDERLINES and ARTICLES.

In order to describe the activity in the data base, one has to know the frequencies of the retrieval processes, and the scattering factors which are a result of transitions between the different record types. A transition from an owner record type to a member record type results in a scattering factor equal to the average number of member records per owner record. It should be obvious that transitions from user to a record type and transitions from a member record type to an owner record type result in a scattering factor equal to one.

The activity in the data base may be represented in a matrix (see Figure 12):

$\{a_{ij}\}$, $i=0,1,\cdots,n$ and $j=1,2,\cdots,n$

where

n is the number of record types

a_{0j} is the number of retrievals of the first kind from user to record type no. j.

a_{ij} $i=1,2,\cdots,n$, $j=1,2,\cdots,n$ and $i \neq j$ is the number of retrievals of the first kind from record type no. i to record type no. j.

a_{jj} is the number of retrievals of the second kind within record type no. j.

$a_{T,j}=\sum_{i=0}^{n} a_{ij}$, $j=1,2,\cdots,n$ is the total number of retrievals of record type no. j.

Suppose the retrieval process S1 is initiated 100 times a day. The example described in Figure 11, then results in the retrieval matrix shown in Figure 13.

The behavior of the retrieval processes will be described with a few more examples. Figure 14 shows the logical model designed in an earlier section. The numbers within the boxes are the numbers of the specific record types and the numbers on the arrows are the scattering factors for transitions from an owner record to the member records. The access paths of four retrieval processes are drawn as dotted lines. The retrieval process S1 has been described earlier, and a description of the remaining processes shown in Figure 14, may be as follows:

S2: Read customer name
Operation: READ
K: —customer no
O: —customer name

From \ To	1	2	⋯	j	⋯	n
0	ω_{01}	ω_{02}				ω_{0n}
1	ω_{11}	ω_{12}				
2						
⋮						
				ω_{ij}		
⋮						
n						ω_{nn}
	$\omega_{T\cdot 1}$	$\omega_{T\cdot 2}$				$\omega_{T\cdot n}$

n is the number of record classes

ω_{0j} is the number of retrievals of the first kind from user to record class no. j.

ω_{ij} $i = 1, 2, \cdots, n, \quad j = 1, 2, \cdots, n \quad \text{and } i \neq j$
is the number of retrievals of the first kind from record class no. i to record class no. j.

ω_{jj} is the number of retrievals of the second kind within record class no. j.

$\omega_{T\cdot j} = \sum_{i=0}^{n} \omega_{ij}, \; j = 1, 2, \cdots, n$ is the total number of retrievals of record class no. j.

Figure 12—Retrieval matrix

The process results in one retrieval of the first kind between USER and CUSTOMERS and the same between CUSTOMERS and CUSTOMERNAMES.

S3: Update quantity
Operation: UPDATE
K: —article no
O: —quantity

The process updates the number of articles in stock, and results in one retrieval of the first kind from USER to ARTICLES.

S4: Remove refill order
Operation: DELETE
K: —refill order no
O: —text string (deletion OK/not OK)

The process removes a refill order and deletes the relationships to ARTICLES and SUPPLIERS. The result is one retrieval of the first kind from USER to REFILL ORDERS, one from REFILL ORDERS to ARTICLES and one from REFILL ORDERS to SUPPLIERS. Suppose that, in addition to the retrieval processes already mentioned, there are other retrieval processes which complete the user's communications with the data base. A possible retrieval matrix may be the one in Figure 15.

One may observe that $\omega_{46}=3990$ and $\omega_{44}=0$. This indicates that "article no" and "article name" also should be stored in ORDERLINES. The decrease in the number of retrievals has to be paid for with an increase of the data volume.

Such a modification of the logical model results in the data structure diagram shown in Figure 16. Further examinations of the retrieval matrix may lead to other modifica-

From \ To	Orders	Orderlines	Articles
User	100		
Orders		100	
Orderlines		600	700
Articles			
Total	Σ	Σ	Σ

Figure 13—The contributions from the retrieval process S1 to a retrieval matrix

tions of the logical model and thus the final solution will be fitted to the selected DBMS in a "best possible" way.

PHYSICAL IMPLEMENTATION

The final logical model describes the final record design and the final access paths. The next step in the design

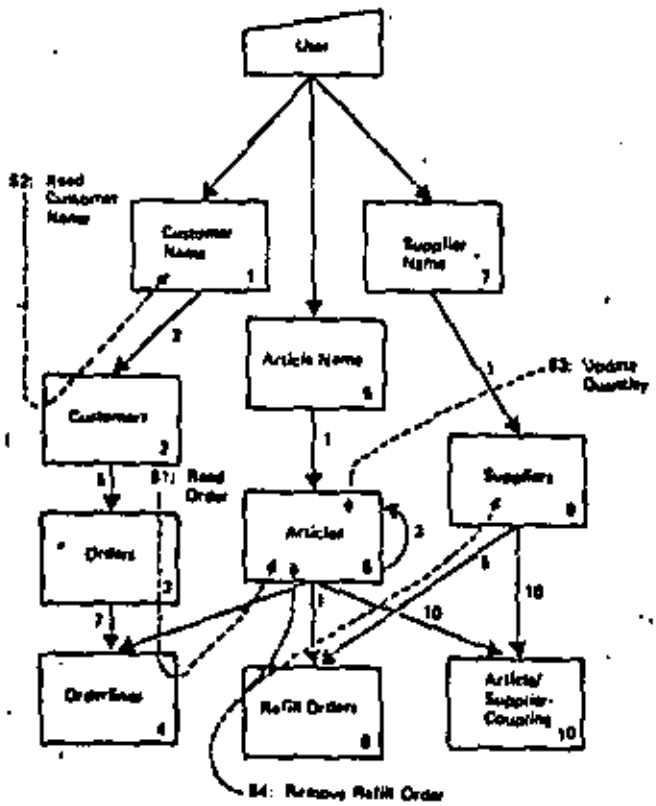


Figure 14

From \ To	1	2	3	4	5	6	7	8	9	10
0	100	100	600		100	400	6	20	11	
1		100								
2	100	100								
3		300		610						
4				3420		3000				
5						100				
6					1536	75		45		130
7									6	
8						30			86	
9							31			1
10						10			1200	1177

Figure 15—Complete retrieval matrix

procedure is a physical implementation of the logical model. This process consists of four main steps:

- choice of storage structures for the record types,
- physical location of the records,
- implementation of the relationships between the records,
- determination of the file dimensions.

The physical implementation is dependent on the particular DBMS and consequently it is not possible to formulate any general rules to solve the problem. Therefore, in this paper,

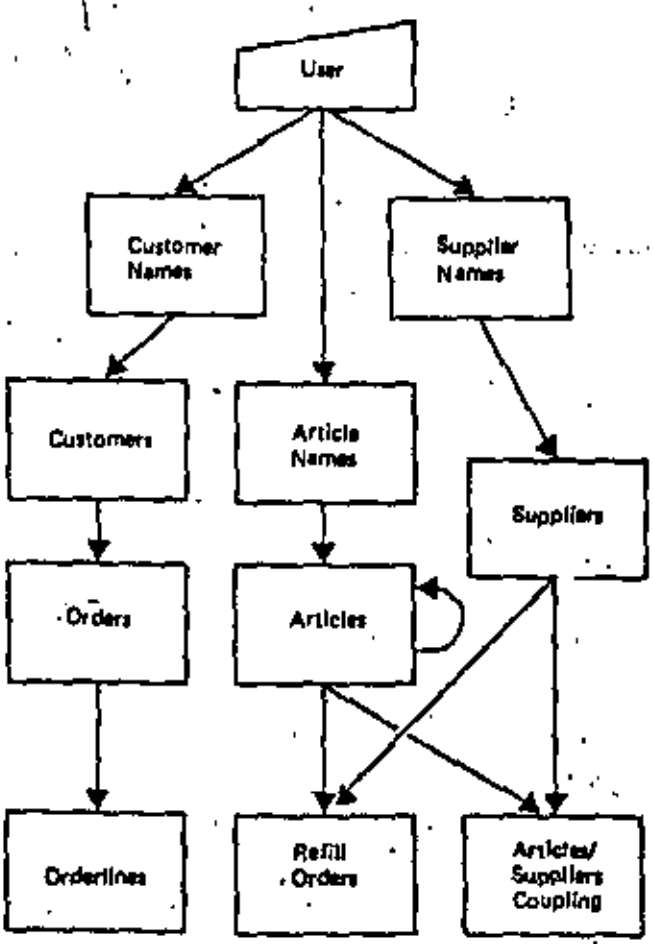


Figure 16

it will be shown how the problems can be solved using a DBTG-DBMS.

Choice of storage structure

A DBTG-DBMS allows the following types of storage structures:

- a direct structure (DIRECT)
- a randomized structure (CALC)
- an index sequential structure (INDEX)
- a sequential structure (VIA)

Usually, there will be no problem in making a choice between these structures. For example a choice between direct access and sequential access is determined by the response time requirements. If direct access is necessary, one should use a direct structured file if possible. If not, an index sequential or randomized structure has to be used. A randomized structure is usually cheaper than an index sequential structure. Therefore, if there is no sequential searching a randomized structure should be used. Of course, these rules are not of general validity, but they may be used in most cases.

Physical location

The physical location may, to some extent, be controlled by the following statements:

(i) $\left\{WITHIN \text{ area-name-1} \left[\left\{\begin{matrix}\text{integer-4}\\ \text{data-base-data-name-4}\end{matrix}\right\} \left\{\begin{matrix}THRU\\ THROUGH\end{matrix}\right\} \left\{\begin{matrix}\text{integer-5}\\ \text{data-base-data-name-5}\end{matrix}\right\}\right]\right\} \ldots$

The statement may be used to specify in what physical data file(s) the various record types are to be placed.

(ii) *LOCATION* MODE IS *VIA* set-name-1 SET
If a sequential record type has more than one owner record type, this statement may be used to place the member records close to one of the owner records.

(iii) *INTERVAL* IS integer-3 PAGES
This statement may be used to place member records close to their owner record, for example in the same physical block.

The retrieval matrix describes the activity between records and it may therefore be a useful tool in determining the physical locations of the records. High activity between certain records implies that they should have physical locations close together.

Implementation of relationships

The implementation of relationships between records (i.e., set types) are done via list structures in a DBTG-DBMS.

A set may be organized as a one-way or two-way list using the statement

MODE IS *CHAIN* [LINKED *PRIOR*]

The choice of list structure has to be based on the description of the retrieval processes. High activity from particular member records to their owner record indicates the use of a pointer from the member records to their owner record. This can be realized using the statement

[LINKED TO *OWNER*]

Determination of the file dimensions

The final step is to determine the dimensions of the data files, i.e., determine the parameters:

- file volumes
- volumes of overflow areas
- volumes of index tables
- load percents
- block sizes

A "best possible" choice of values for the parameters mentioned above is dependent on the particular DBMS and the particular hardware used. Consequently, such decisions should be made in connection with a performance analysis. This will be further discussed in the next section.

PERFORMANCE ANALYSIS

An evaluation of the final solution should be done using a performance analysis. The most convenient criterion of a good solution is low total cost per time. Various types of costs which affect the choice of file organization are:

- storage costs
- access costs
- data transfer costs
- sorting costs
- reorganization costs

In addition there will be costs for using cpu and memory.

In order to get a better picture of the performance analysis procedure, a simplified case, the one-identifier case, will be considered. In the one-identifier case the entire file organization consists of a single file with records consisting of one identifier item and one or more property items.

In order to develop a cost function, it is necessary to have information about

- the retrieval activity
- the storage structure and the dimensions of the file
- the accounting routine of the particular machine system

The first step is to decide on a storage structure. For a particular storage structure one may decide on block size and load percent. Those decisions will usually be results of local optimizations, depending on the particular computer

system that is used. In addition, it is necessary to estimate the volume of internal housekeeping data in the file. Thus, the file may be described by the following parameters:

- number of records
- record size
- block size
- load percent
- size of overflow-area
- volume of housekeeping data

Based on this description, one may estimate the average retrieval length (measured in block accesses). How this can be done for the most common used storage structures is described in Reference 3. Estimates of the average use of memory and CPU for each retrieval process must also be made. The retrieval processes are described with the following parameters:

- frequencies
- use of memory
- use of CPU
- retrieval lengths

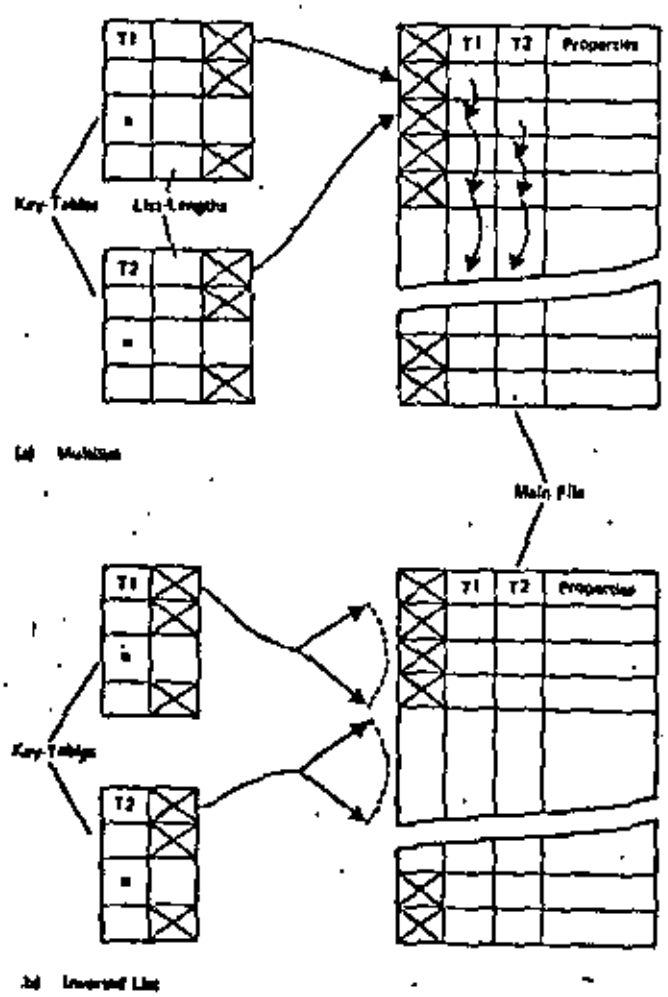


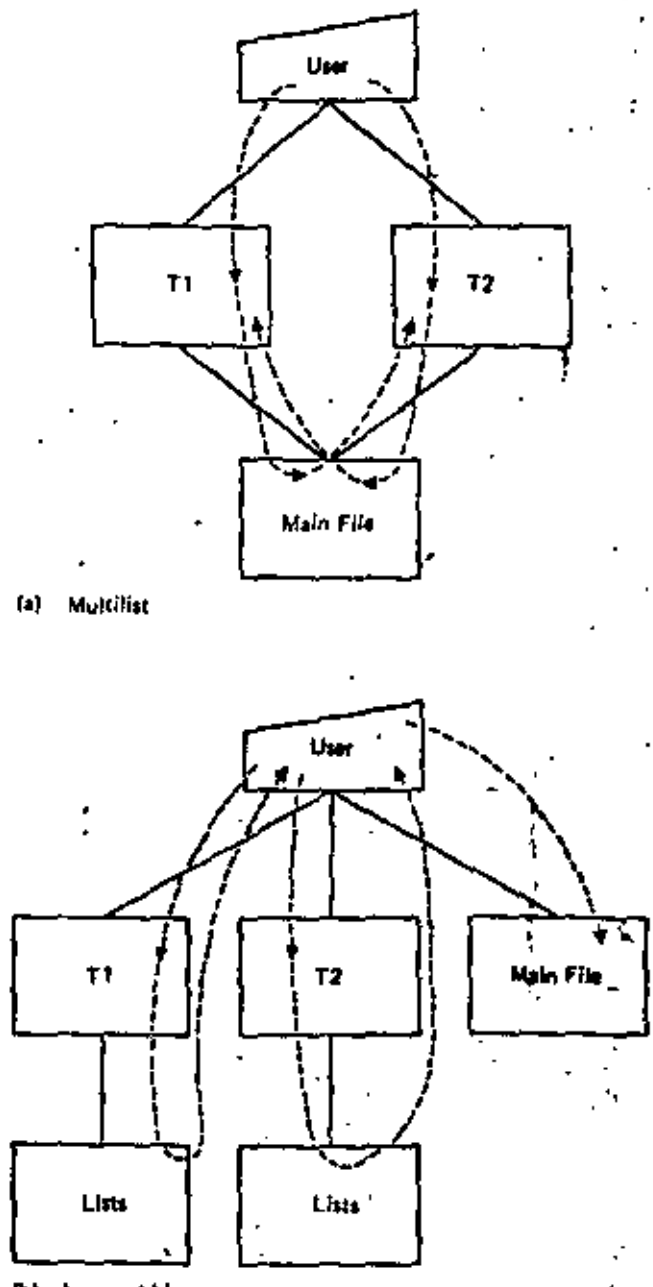
Figure 17—Multilist and inverted list

The parameters mentioned above are used as parameters in the accounting routine of a particular machine system, and a price may be computed.

It is then possible to adjust the file parameters, for example the block size, and look at how these adjustments influence the total price. Thus, a minimization of the cost function may be done in an iterative way.

Searching in the multi-identifier case means that logical conjunctions between the separate identifier items have to be done. The most common ways to implement conjunctions are multilists or inverted lists.

Figure 17(a) illustrates a multilist. T1 and T2 are terms which are parts of the identifier. A multilist consists of key-



(a) Multilist

(b) Inverted List

Figure 18—Multilist and inverted list in a data structure diagram

tables and a main file. A key-table consists of one record for each key value; the property parts of the records contain a list address and a list length. The search method is to find the list addresses in the key-tables and then search through the shortest list. The retrieval length will be the number of accesses to the key tables plus the number of accesses to the main file.

Figure 17(b) illustrates an inverted list. An inverted list consists of key tables and a main file. The result of searching in a key table is a record of variable length which contains all the addresses of the records in the main file having specified key value. The search method is to find the record addresses in the key-tables; then logical functions are computed in memory, and the remaining record addresses are found in the main file. The retrieval length will then be the number of accesses to the key tables plus the number of accesses to the main file.

These list structures consist of separate one-identifier-case problems—searching in key-tables and searching in the main file. Therefore, it is possible to develop a cost function using the same parameters as for the one-identifier case.

Fig 18 describes the multilist and the inverted list in a data structure diagram. The access paths are drawn with dotted lines. From data structures of the kinds shown in Figure 18, it is possible to construct any complex data structure, and any complex data structure can be decomposed into simple list structures. It should then be obvious that a performance analysis of a multi-identifier case may be performed using the same parameters as for the one-identifier case. One should, however, notice that the access paths of the retrieval processes have to be described, and that the retrieval length, in the multi-identifier case, may be written as:

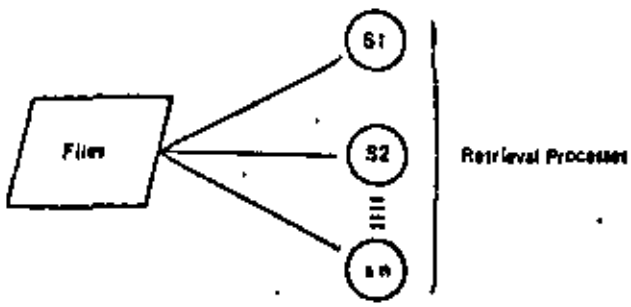
$$a = a_1 + a_2$$

where

a_1 = the number of block accesses caused by the retrieval keys

a_2 = the number of block accesses caused by the establishing of relationships

Figure 19 gives a summary of the necessary parameters.



(i) Retrieval Process Parameters:
- Frequencies
- Retrieval Lengths
- Use of CORE
- Use of CPU

(ii) File Parameters:
- Number of Records
- Record Volume
- Block Size
- Load Per cent
- Volume of Administration Data

(iii) Parameters Which Describe a Particular Accounting Routine

Figure 19—Necessary parameters to carry out a performance analysis

CONCLUSION

In the previous sections a multi-level design procedure for design of file organizations has been developed. The design procedure was based on an information analysis of the application system. The information flow was described using the principle of hierarchical systems partitioning, and necessary information was extracted from the information analysis:

- description of objects and relationships between objects,
- description of permanent files, i.e., message types,
- description of retrieval processes.

A logical model was now developed in three steps:

- the message types of the permanent files were normalized,
- a synthesis of the normalized message types into a logical model was made,
- the model was modified in order to satisfy the retrieval requirements.

So far no decisions about any DBMS have been made. The logical model was now modified to fit a particular DBMS in a "best possible" way, i.e., modifying the record layouts and altering the access paths. The record layout was modified making a selection between duplicate storing of data or the use of a reference, and the access paths were altered by making selections between lists and inverted lists.

Therefore, a physical realization in four steps was done, i.e.:

- choice of storage structure,
- physical location of the records,
- implementation of the relationships between the records,
- determination of the file dimension.

The file dimensioning was done in connection with a performance analysis.

The design procedure outlined in this paper needs a detailed description of the various levels, and further work has to be done in this area.

An experience during the work already done was the strong need for computer based tools for documenting information structures etc., and for analysis of DBMS-schema performance properties. The basis for such a computer-based tool is an automatic documentation of the information analysis. Such documentation systems exist today, for example, CASCADE, which with a few minor adjustments would be well fitted for this purpose. A transformation of the application system into a logical data base model will be the next step. It seems that the method outlined in this paper may be a useful basis for such a transformation algorithm.

Concerning the modification of the logical model, an analysis tool has to be developed. With the information [illegible] ned through the analysis of the application system it seems possible to develop an analysis tool based upon activity between the records.

Finally, the physical realization using a particular DBMS has to be done. It seems that this may be done in connection with a computerized performance analysis.

Considering the obvious possibilities of automating the design procedure, there seems to be good hope of achieving good results in this area in the future.

REFERENCES

1. Codd, E. F., "Normalized Data Base Structure: A Brief Tutorial," 1971 ACM SigFidet Workshop, *Data Description, Access and Control*, edited by E. F. Codd and A. L. Dean.
2. Langefors, B., *Theoretical Analysis of Information Systems*, Third Edition, Studentlitteratur, Lund, 1971.
3. Bratbergsengen, Hafstad, Wiker, "Filsystemer og databaser."

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

OPERATIONAL SOFTWARE FOR RESTRUCTURING NETWORK DATABASES

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

Operational software for restructuring network databases*

by DONALD E. SWARTWOUT, MARK E. DEPPE and JAMES P. FRY
The University of Michigan,
Ann Arbor, Michigan

ABSTRACT

A high-level "access path" approach to database restructuring is described and contrasted with the "elementary operations" approach taken by most restructuring systems. With the elementary operations approach, restructuring is viewed as a sequence of basic or "primitive" operations which manipulate a source database in order to convert it into a target database. In the access path approach, restructuring is seen as the process of accessing a body of information represented by the source data, and constructing the target database representation of the same information. While the elementary operations approach is useful for restructuring hierarchical databases, it does not generalize well for networks. The access path approach is better-suited to the complex structures possible in network databases.

The access path approach permits the specification of complex restructuring transformations in terms of application-oriented concepts such as access strategies and selection criteria. A non-procedural Network Restructuring Language (NRL) based on this approach is presented, and an example of its use in restructuring is given. The architecture of an NRL-driven Restructurer for network databases is described.

INTRODUCTION

Due to the fact that the database design process is essentially an opaque art,[1] an important tool needed by a Database Administrator is a generalized reorganization facility. With such a facility the designer can adapt existing database structure to conform to new information or processing requirements. For example, a new entity or relationship could be added to the database structure to satisfy a new information requirement, or a new access path could be added to satisfy a processing requirement.

Although some reorganization schemes exist in database management systems, they deal primarily with the physical organization of data, e.g., performing garbage collection, changing the blocking factors, and to some extent, modifying the accessing mechanism. To our knowledge no generalized reorganization facility exists in any database management system and, in particular, we are unaware of a restructuring facility which would provide the capability to change the logical structure of the database.

In the current state of the art, the reorganization facility has developed only within the context of a Data Translator.[2–6] As described in References 2 and 6, a Data Translator is a generalized software system that accepts as input a source database, descriptions of the source (input), target (output) databases and the mapping between the source and target logical structures. Its output is the reorganized target database produced automatically by the generalized software from the input description. The execution of the translator invokes three major modules—Reader, Restructurer, and Writer. While the function of the Reader and Writer modules can be classified primarily as physical reorganization or *reformatting*, the primary purpose of the Restructurer module is to logically reorganize the database or restructure it.[3] Bims and Fry[6] described the physical reformatting capabilities of the Reader and Writer through the logical restructuring capabilities which were embodied in several prototype implementations. In this paper we focus on the development of a generalized restructuring capability for the network class of logical structures.

What is restructuring?

The purpose of restructuring is to transform the logical structure of a database in response to new information or processing requirements. Navathe and Fry[7] developed a categorization of restructuring operations for the hierarchical class of logical structures. Three fundamental logical structure modifications were defined—Naming, Relation, and Combining—which were refined into several lower-level restructuring operations. Figure 1 illustrates a few of these hierarchical restructuring operations. The first part, Figure 1a depicts a hierarchical relationship among PRESIDENTS, SPOUSES and CHILDREN. The uppermost box

* Supported by WWMCCS ADP Directorate Defense Communications Agency Contract No. DCA-100-75-C-0064

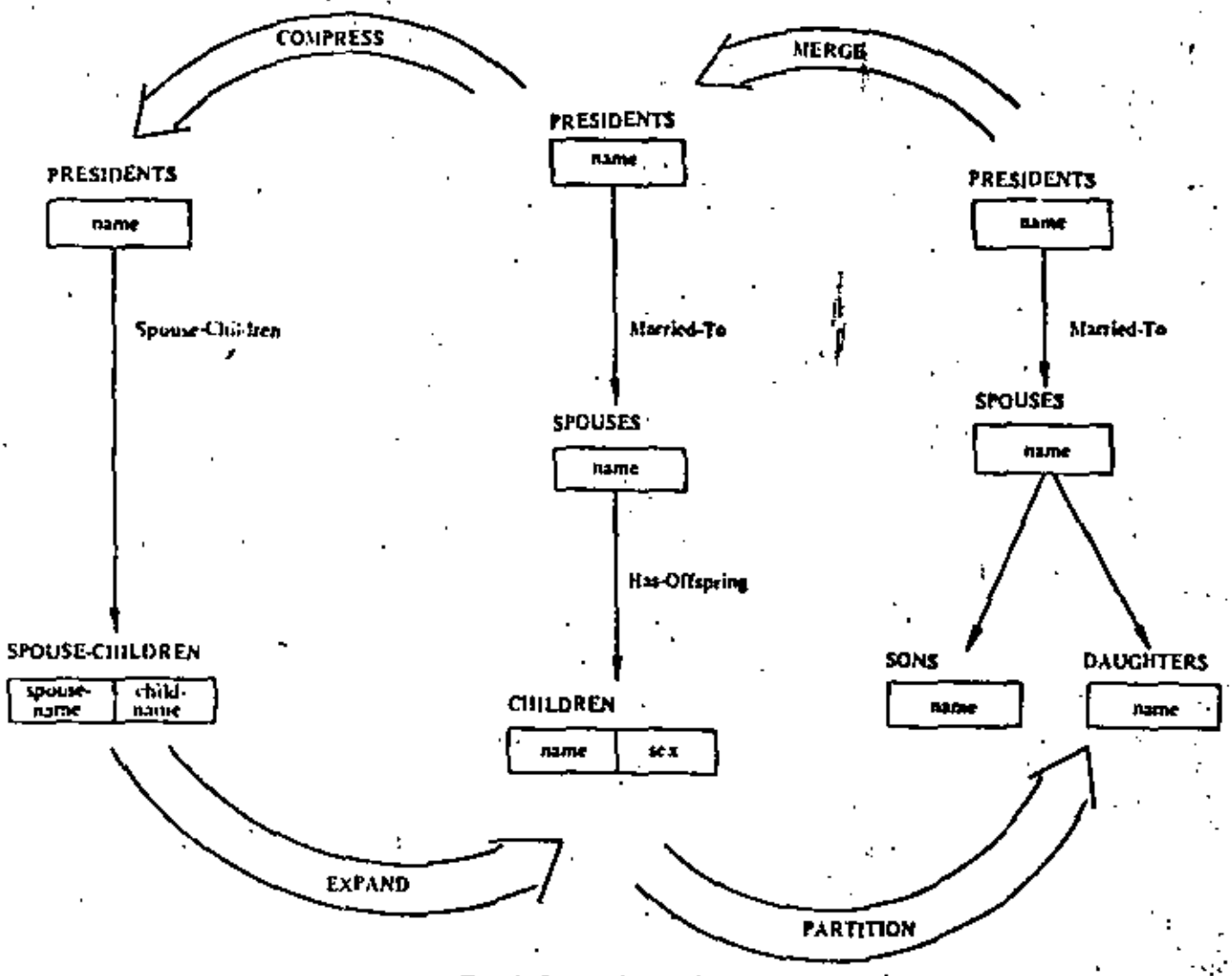


Figure 1—Restructuring operations

or record type,** PRESIDENTS,† contains the names of the presidents of the United States. The second level record type, SPOUSES, is related to PRESIDENTS through the set type "Married to" and contains the names of all the spouses for each president. Thus, the set is a one-to-many mapping over two record types. In a similar fashion, the third level record type, CHILDREN, is related to SPOUSES and contains the names of the children born to each SPOUSE.

One restructuring operation is *partitioning* in which the occurrences of a record type are divided into two or more distinct record types based upon the value of one or more data items. This operation is illustrated by the source and target logical structure of Figure 1 where the source record type CHILDREN has been partitioned into two record types, SONS and DAUGHTERS, based on the data item "sex." Notice that the data item "sex" has been eliminated in the logical structure of Figure 1, since this information is now represented by the target logical structure. The dual restructuring operation of partitioning is *merging*. The merging operation consolidates occurrences from two or more record types into a single record type, often adding a data item to the record type to preserve information previously represented by the logical structure. Merging is illustrated by the source logical structure and the target logical structure of Figure 1, where the occurrences of the two record types, SONS and DAUGHTERS, have been merged into a single record type, CHILDREN. Note that the terms "source" and "target" are relative and depend on the direction of the restructuring transformation. Consequently, Figure 1 represents the source logical structure for the partitioning example, and the target database for the

** We follow the terminology of DBTG.[9]

† For a more complete description of the Presidential Database see Reference 10.

merging example. Notice the item type "sex" has been added to contain the information previously represented semantically by the source logical structure of Figure 1. Another form of restructuring involves the *compression* of two or more hierarchical levels into one. This example is illustrated by the source and target logical structure of Figure 1 where the two source records, SPOUSES and CHILDREN, have been compressed into a single record type, SPOUSE-CHILDREN. On the record occurrence level, this operation would be accomplished by replicating the associated SPOUSE record occurrence for every CHILD record occurrence. The dual operation, *expansion* expands one level of hierarchy into two or more by factoring out selected data items. This procedure is illustrated in Figure 1 where the SPOUSE-CHILDREN record type has been factored into two levels.

Network restructuring

Although a complete categorization of such operations has been established for the hierarchical class of structures,[4] restructuring operations are not nearly as easy to classify in the network class of logical structures due primarily to the variety and complexity of the structures involved. For example, an important network restructuring operation, *changing the implementation of many-to-many relationships*, is illustrated by the restructuring transform of Figure 2. The source logical structure represents information on students, their class standing, all courses taken by each student, and the final grade received in each course. The target logical structure represents exactly the same information except that the association between students, course, and grades is achieved using a LINK record type. This example is typical of the restructuring operations which may be posed in a network environment, those which involve several record and set types.

Another example, which might enhance processing, would be the *addition of indexing sets or record types*. This could be achieved by migrating the data item "class" to the record type CLASS-STANDING, which would serve to segment the students into graduate, undergraduate, special, foreign exchange, etc. (Figure 3). Notice that this operation is actually expansion performed on a hierarchical substructure of the target structure of Figure 3, namely the single record type STUDENTS.

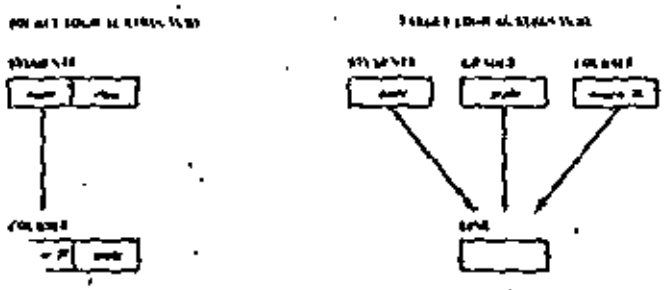


Figure 2—Network restructuring example

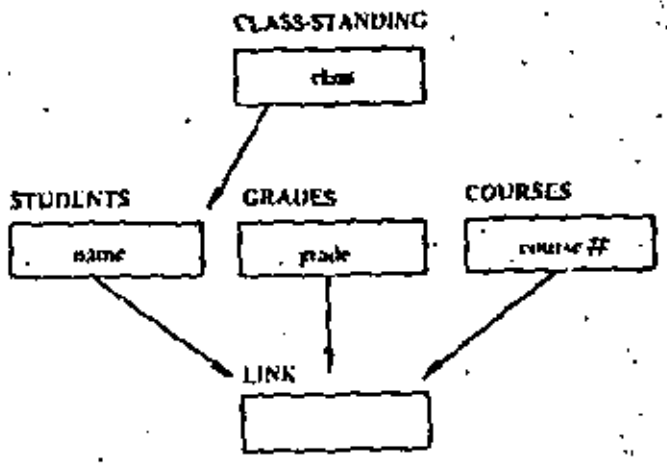


Figure 3—Addition of indexing sets

Yet another important network restructuring capability is the ability of the restructuring process to extract not only data which exists explicitly in the source database, but also that which exists implicitly, i.e., the information which may be inferred from the actual source data. An example of the difference between implicit and explicit information is described in the hypothetical restructuring transform of Figure 4.

The example describes a source database containing two record types, PERSONS and LINK. The PERSONS record type contains name and sex item types, while the LINK record type contains no data but provides relationship information in conjunction with sets PARENTS and CHILDREN. The target structure also contains the PERSONS record type, and construction of target PERSONS records involves the extraction of data explicitly resident in the source file PERSONS record type. In contrast, target record types PARENTS and GRANDFATHERS do not correspond directly to any source file record type; they contain information which is represented implicitly in the source file. Needless to say, this transformation cannot be described by operations on hierarchical substructures.

In the next section, we review the previous work in developing hierarchical restructuring capabilities through specification of elementary operations. The third section comprises a discussion of the access path approach necessary to achieve a network restructuring capability, and the fourth section describes a Network Restructuring Language based on this approach. The implementation of the Restructurer is discussed in the fifth section and the paper concludes with our observations on building and using restructurers.

THE ELEMENTARY OPERATIONS APPROACH

An interesting analogy exists between restructuring systems and high-level query systems. In fact, one can consider a query as a restricted restructuring transformation in which the target is not a database but some other form of

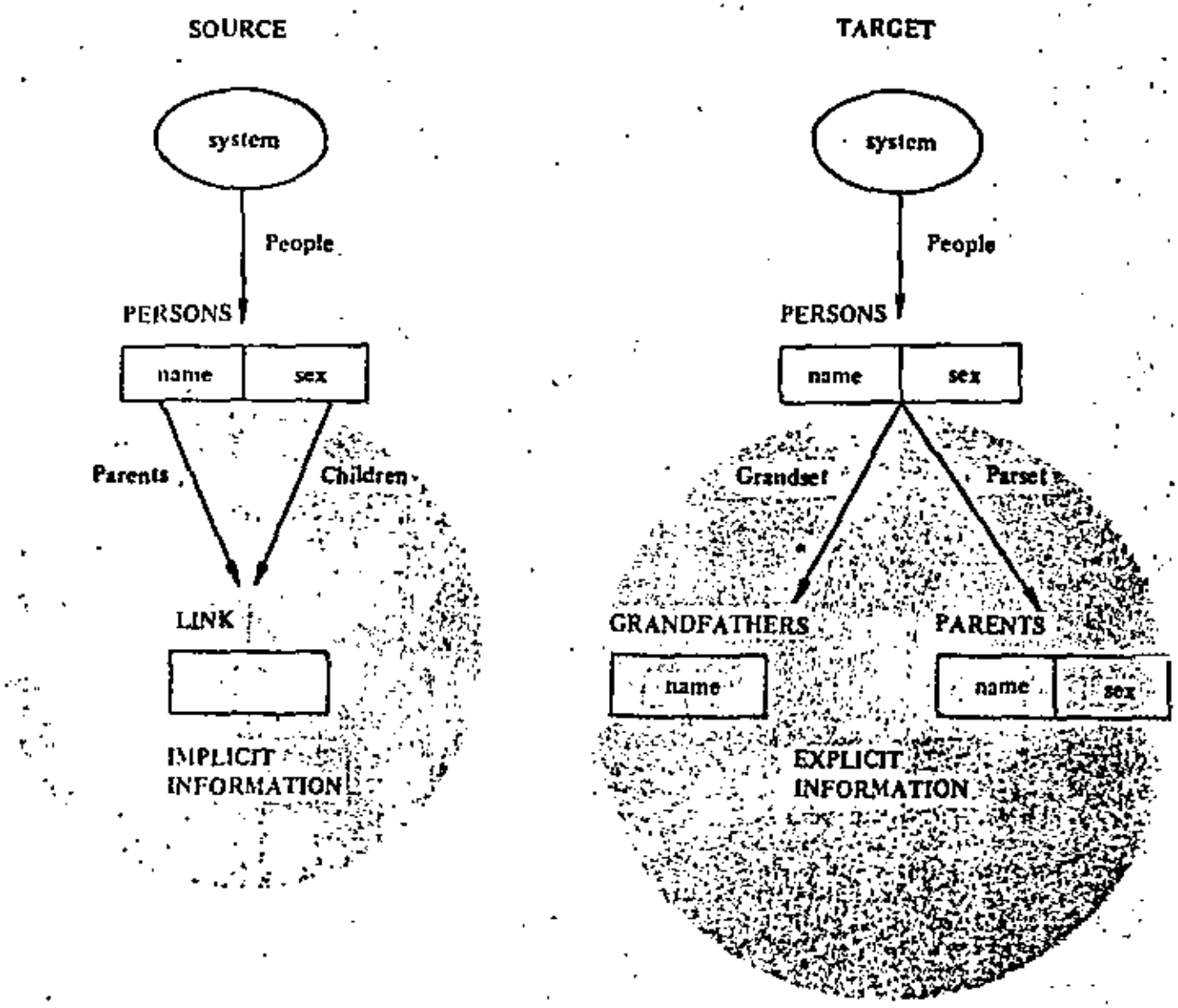


Figure 4—Implicit data extraction

information representation. Efficient, easy-to-use query languages for the hierarchical class of logical structures have been built through the use of elementary operations and have been in existence for some time.[9] These systems allow a user to specify a sequence of elementary operations on hierarchical structures which will accomplish his query. Such languages are easy to learn and use since they employ a few rather simple operations to perform most queries to hierarchical structures.

Several research efforts have addressed the problems of restructuring and restructuring language specification from the elementary operations point of view. CONVERT, a high-level translation language, provides a generalized restructuring capability for hierarchical structures. The approach is based upon the concept of a "data form" in conjunction with a set of restructuring functions called "form operations." Shoshani[11] takes a similar approach to restructuring whereby a set of "conversion functions" is used to specify restructuring operations.

In the elementary operations approach, the source database is considered to be a collection of data in a specific logical structure and format, while the target database is viewed as essentially the same data, but in a different logical structure and format. Restructuring, therefore, is considered to be the process of manipulating the source data to conform to the target logical structure and corresponding format. Consequently, restructuring research using this approach turns toward the development of low-level or "primitive" operations which transform occurrences of one logical structure to another. One advantage of such an approach is that the architecture of the restructuring software system is greatly simplified: it defaults to a set of low-level subroutines which correspond directly to the elementary operations. Thus, a restructuring specification

consists of a sequence of primitives which may be directly converted to a sequence of subroutine calls which perform the actual restructuring. Unfortunately, the user is not shielded from any aspects of the restructuring: he must thoroughly understand the function of each operation in order to be able to use it. Consequently, the language, although high-level, still requires the user to treat restructuring essentially as a sequence of low-level steps.

Several other problems arise when the elementary operations approach is applied to restructuring network databases. In general, elementary operations are designed to operate on small, logical substructures consisting of one or two record types and a set, to produce a new, logical substructure. Due to the limited number of operations which may be defined, only a finite number of source substructures can be valid candidates for restructuring. The more complex the structure, the greater the probability that it will contain substructures which are not valid candidates for the set of available operations. Consequently, the restructuring capabilities of elementary operations decrease as the complexity of the structure increases. For this reason, elementary operations are not particularly well-suited to describing restructuring transformation upon complex network logical structures. In addition, a complex restructuring transformation (assuming that it may be performed by the elementary operations) will require the specification of a complex sequence of elementary operations which is difficult to analyze and to understand.

Another class of restructuring transformations that is difficult to accomplish with elementary operations is the extraction of implicit information. We have pointed out that such transformations (as in the example of Figure 4) cannot usually be described by a sequence of operations on hierarchical substructures. Furthermore, although they may be describable by sequences of elementary operations on network substructures, such descriptions are generally long and complicated. There is also reason to believe that, given time, designers of network databases will produce enough intricate methods of storing information implicitly to exhaust any set of elementary operations.

Following our analogy, it has been found to be extremely difficult to generalize the elementary operations of high-level query systems to network databases.[4] The elementary operations approach to developing a query system for network databases tends to be less powerful and much more cumbersome than its hierarchical counterparts. In general, they suffer from the same problems cited above—limited allowable input structures, overly complex specifications, and difficulties with link records and other implicit information storage techniques. The high-level access path approach to restructuring grew out of attempts to develop a restructuring strategy more suited to network databases.

THE ACCESS PATH APPROACH

Following the current trend in host language database systems which process the network structure databases, we chose the high-level "access path" approach to restructuring. The source database is viewed as a body of information, some of which is represented explicitly by data, and some of which is represented implicitly, i.e., may be inferred from the data. Similarly, the target database is considered to contain a subset of information represented by the source; this data is created from information provided by the source data. Executing a restructuring transformation, then, is simply the process of traversing the source database to obtain the information needed to create the target database, and storing it according to the target logical structure. There are numerous consequences of this approach. For one, research in restructuring specification turns toward the development of restructuring specification languages based on the concepts of access strategies and selection criteria. Since this area is closely related to query language development, certain concepts from previous research in this area may be utilized. A second consequence of the access path approach is manifest in the actual restructuring algorithm development. Restructuring technology turns toward the development of algorithms which efficiently and exhaustively access the source database and perform tests upon the data, following externally specified access strategies and test criteria. This is a radical change from the elementary operations approach which tends to develop low-level subroutines.

The most important consequence of this approach is that it produces powerful generalized network restructuring capabilities. Since restructuring is viewed as an operation which accesses information (rather than manipulates data), the approach is unaffected by the logical structure of the database. This independence from the logical structure insures that essentially any database is a valid candidate for restructuring, regardless of the complexity of the logical structure (hierarchical, network, etc.). Furthermore, source and target logical structures need not even remotely resemble each other since the target is derived from information provided by the source rather than from the source structure itself. Also (unlike the elementary operations approach), implicit information may be extracted and restructured as easily as explicit information (as in Figure 4). Thus, explicit information may become implicit and vice versa.

Finally, the system is inherently simple. Assuming that the user has some familiarity with databases and applications, a restructuring specification language based on application-oriented concepts such as access strategies and selection criteria should be easily understood. Furthermore, since all restructuring operations are expressed as information accessing problems, confusion does not significantly increase as the complexity of the restructuring transformation increases. An algorithm designed to carry out this process is also simple and therefore, straightforward and dependable. All restructuring is performed in an identical sequence of steps, regardless of the particular transformation: (1) exhaustively access the source data according to the access specifications, (2) test data based on selection criteria, and (3) create target record occurrences containing the relevant data. For a more complete discussion of the theoretical foundations of this approach, see Reference 13.

NETWORK RESTRUCTURING LANGUAGE

Architecture

The architecture of the Network Restructuring Language is based upon the high-level specification of access paths. The major components of the language describe access strategies and selection criteria. Access strategies are described using an access path statement which specifies the traversal scheme required to obtain data for each target record type. The selection criteria establishes the source (and, indirectly the target) data requirements. It would be beyond the scope of this paper to describe the NRL in great detail. A complete language specification may be found in Reference 14. However, the major points of the language will be described.

The NRL is essentially a block-structured language in which each "block" contains a single *target record statement*. There is one target record statement for each record type in the target database. This structure reflects the access path approach for the following reason: each target record type is considered to represent a certain quantity of information which may be obtained from the source database; consequently, each target record description contains the specifications for the source database accessing scheme as well as the selection criteria. The second level of structure in the NRL is the *target set statement* (see Figure 5 for the NRL structure). Each target record statement includes one or more target set statements which identify the sets of which the target record type is a member. The third level of NRL structure is the *access path statement*. The access path statement specifies exactly how the source logical structure is to be traversed in order to obtain the information necessary to create an occurrence of the target record type. There may be many individual access path statements—one or more for each target set statement—since the information which contributes to the target record type may come from several different source record types.

```
TARGET RECORD STATEMENT
  TARGET SET STATEMENT
    ACCESS PATH STATEMENT
      NEW TARGET ITEM STATEMENT
      SOURCE RECORD STATEMENT
        ITEM QUALIFICATION STATEMENT
        ITEM ASSIGNMENT STATEMENT
          •
          •
      SOURCE RECORD STATEMENT
          •
          •
    ACCESS PATH STATEMENT
      •
      •
  TARGET SET STATEMENT
    •
    •
TARGET RECORD STATEMENT
  •
  •
```

Figure 5—Structure of NRL specification

There are two types of NRL statements at the fourth level of structure: the *new target item statement*, and the *source record statement*. There may be zero or more new target item statements for each access path statement. Such a statement indicates a target item which receives a constant value each time a target record occurrence is constructed using the specified access path. Since access paths indicate structure in the source database, the new item statement is useful when information represented semantically in the source structure is converted to actual data values in the target structure.

The second NRL statement at the fourth level of structure is the *source record statement*. The source record statement identifies the source record(s) on the access path which will be used in the creation of a target record occurrence, to obtain item values and/or test the source data. There may be one or more source record statements for each access path statement since data may be obtained or examined from several different source record types. Furthermore, each source record type may be optionally assigned an *index number*. The index number is used to identify uniquely a source record within an access path which "cycles" or "loops" back on itself such that the same source record type appears more than once. Finally, there are two statement types at the fifth and final level of NRL structure; they are called the *item qualification statement*, and the *item assignment statement*. Item qualification statements are used to establish the selection criteria used in testing the source data. Consequently, one may specify a constant value to be compared against a source item value which will determine whether or not a target record occurrence is to be created using the current set of source data. The item assignment statement is used actually to obtain the source item values and assign them to the proper target items.

It should be noted that the NRL does not explicitly describe the logical structure of either the source or target database. The Michigan Data Translator obtains appropriate descriptions of the source and target databases independently of the restructuring specifications. The NRL, therefore, describes only the information relevant to the actual transformation from source to target, and assumes that the logical structures of the databases have been previously defined and are available to the Restructurer.

An NRL restructuring example

The NRL example has been chosen to illustrate the capability of the language to specify restructuring capabilities which may not be readily classified in terms of hierarchical structures or elementary operations. The example involves the extraction of implicit information from a network structure and is taken from Figure 4. This example is reproduced in Figure 6 for convenience. The NRL required to accomplish the necessary transformation is documented in Figure 7. The form of the NRL has been simplified slightly for clarity. The block structure of the language may be easily observed. Because there are three target record

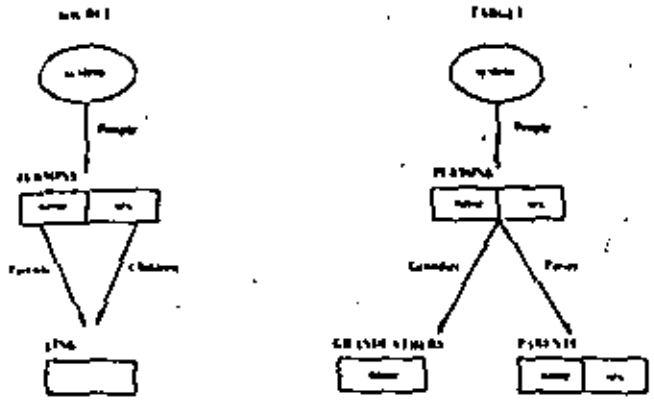


Figure 6—Restructuring example

types, there are three target record statements. Statement #2 begins the NRL description for target record type PERSONS. Since PERSONS is a member of only one set (from the system access level) there is only one target set statement (#3). Statement #4 specifies the source access path to be used to locate the data necessary to create the target record type. ACCESS PATH=PEOPLE (PERSONS) describes an access path from the source SYSTEM node along set PEOPLE to record type PERSONS. Statement #5 is the source record statement to indicate that source record type PERSONS is to be used to obtain information. Statements #6 and #7 are item assignment statements which indicate that the values in source item types "name" and "sex" are to be assigned to the target item types "name" and "sex."

The NRL description for the second record type, GRANDFATHERS, begins with statement #8. As before, there is only one target set statement (#9) because the record type is a member of only one set (GRANDSET). Statements #10, #11, and #12 specify the source access path to be used to obtain the desired information. This access strategy may be summarized as follows. The LINK record type in the source models the relation between parents and children. Given a particular PERSONS record occurrence, one may access all of his/her parents by using the set types PARENTS and CHILDREN in conjunction with the LINK record type. One similarly obtains all grandparents of a person by accessing all parents of parents. The access path statement on lines #10, #11, and #12 describes the essence of this strategy: use set PEOPLE to the PERSONS record type, then set PARENTS to the LINK record type, then set CHILDREN back to the PERSONS record type, then set PARENTS back to the LINK record type, and finally, set CHILDREN back to the PERSONS record type. Notice that the source record type PERSONS is used three times in the access path specification: once as a person, once as a parent, and once as a grandparent. Consequently, the source record statement (#13) must indicate which source group PERSONS is to be used. Since the PERSONS record which represents grandparents is the fifth record on the access path, INDEX=5 makes the necessary distinction. Statement #14 is an item qualification statement which is used to select only male grandparents to yield grandfathers, the desired target record information. Statement #15 assigns the value in the source item type "name" to the target item type "name."

It should be clear that the target record type PARENTS

```
1)  NRL
2)      TARGET RECORD PERSONS
3)          TARGET SET PEOPLE
4)              ACCESS PATH = PEOPLE (PERSONS)
5)                  SOURCE RECORD PERSONS
6)                      NAME = NAME
7)                      SEX = SEX

8)      TARGET RECORD GRANDFATHERS
9)          TARGET SET GRANDSET
10)             ACCESS PATH = PEOPLE (PERSONS), PARENTS (LINK),
11)                           CHILDREN (PERSONS), PARENTS (LINK),
12)                           CHILDREN (PERSONS).
13)             SOURCE RECORD PERSONS, INDEX = 5
14)                 SEX QUALIFY IF = 'MALE'
15)                 NAME = NAME

16)     TARGET RECORD PARENTS
17)         TARGET SET PARSET
18)             ACCESS PATH = PEOPLE (PERSONS), PARENTS (LINK),
19)                           CHILDREN (PERSONS).
20)             SOURCE RECORD PERSONS, INDEX = 3
21)                 NAME = NAME
22)                 SEX = SEX
23) END NRL
```

Figure 7—NRL specification

is created in a manner similar to that of GRANDFATHERS, except that the accessing strategy is simpler and no selection criterion is required.

OPERATIONAL SOFTWARE

There are basically two approaches to implementing the modules of a data translator: generative and interpretive. In the generative approach each module generates an object program which, when executed, performs the desired function of the data translator. The advantage of this approach is that efficient machine code may be generated for each application or function. This efficiency may be irrelevant, however, since transformations are rarely executed more than once. The disadvantage of this approach is that two phases (code generation and module execution) are required to arrive at the final result.

The interpretive approach consists of a table-driven program that is applicable to all potential transformations. The disadvantage of this approach is the possible inefficiency of a general purpose interpreter. However, the program is easier to understand and debug.

Restructurer design decisions

Overall, the basic design objective of the Michigan Data Translator is to provide an operational software system for demonstrating, testing, and validating research results on generalized data translators. The primary design goal for the Restructurer module is the incorporation of general network restructuring capabilities.[14] Due to the prototype nature of the task, and to reduce the implementation time and enhance the experimentation and validation effort, the simplest and most straightforward algorithm was selected. Consequently, little emphasis was placed on execution efficiency. Efficiency development was left to future versions of the Michigan Data Translator.

The decision to use an internal DBMS as an implementation tool was made early in the design process. It reflects the need for an environment in which system tables are shared by translator modules, and change often in size and complexity. In addition, since the Michigan Data Translator was developed in a research environment, the table designs themselves were subject to change. Therefore, some degree of centralization, integrity control, and data independence for system tables was clearly necessary. The translator uses *ADBMS*, a DBTG-type DBMS developed at the University of Michigan by the ISDOS Project.[15] All databases, except the source and target databases, are ADBMS databases. They are manipulated by ADBMS "verbs" which take the form of subroutine calls.

Another design decision to provide generality and to ease the complexity of the Restructurer's implementation resulted in the development and use of an internal data form.[2,12,13] As summarized earlier, the Reader module in the Michigan Data Translator accesses the source database and converts it into the internal format called the source Restructurer Internal Form—source RIF. In addition to maintaining the logical structure of the source database, the RIF database has additional system access sets to every record type which facilitates fast access by the Restructurer.

Restructurer algorithm

The function of the Restructurer algorithm is to provide all the restructuring capabilities discussed in the first part of this paper. Since the Reader and Writer modules isolate the Restructurer from the source and target databases, the Restructurer need be tailored only to handle RIF databases. The Restructurer, then, is essentially an ADBMS-specific data translator and is directed by the contents of the NRL tables. The basic cycle of the Restructurer is divided into five phases as indicated in Figure 8.

During the first phase, Target Control, the Restructurer determines the next step in its traversal of the target logical structure. That is, a target set/record pair is selected for construction, as is a current access path to direct the construction process. The paths that the Restructurer takes in traversing the target database are called "construction paths." They are defined implicitly by the TARGET SET and TARGET RECORD statements supplied by the user in his NRL specification.

During the Source Accessing phase, the Restructurer

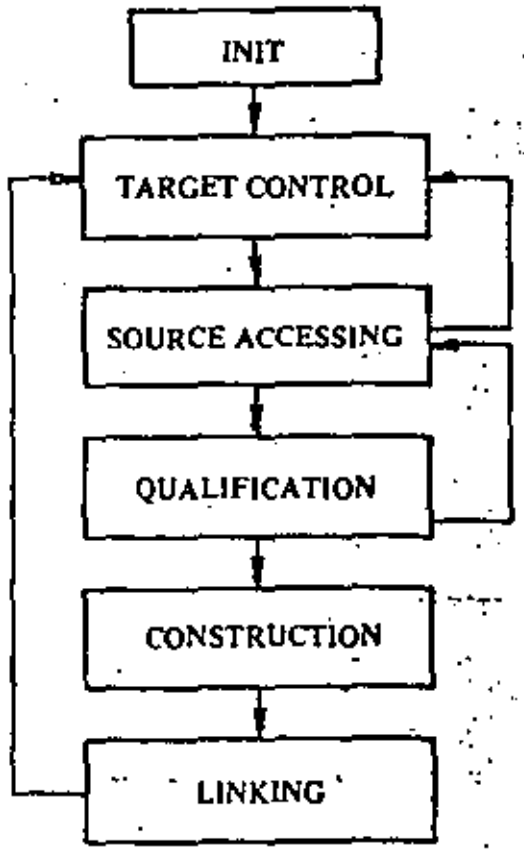


Figure 8—Restructurer block diagram

determines the next step in its traversal of the source RIF, as indicated by the current access path. In essence, an access path is a continuous list of source relations which indicates the location in the source of information required to create a target instance.

The purpose of Source Accessing is to exhaustively search for every potential target record occurrence that could be created from the records on the source access path. When a new record occurrence is found, the Restructurer enters the Qualification phase. If none is found, that is, when the access path has been exhaustively examined, the Restructurer returns to the Target Control phase.

It is during the Qualification phase that the Restructurer implements the user's "selection criteria." The source record occurrences on the current source access path are retrieved and all specific items to be qualified are tested against the values specified in the NRL. If *all* of the occurrences pass, the Restructurer enters the Construction phase. Otherwise, the Restructurer returns to the Source Accessing phase to find another access path.

During the Construction phase, source record occurrences along the current access path are retrieved. Data items are extracted, conversions are performed on them as necessary, and they are stored into a new target record occurrence.

The Linking phase is the most complex phase in the Restructurer cycle. Basically, its job is to connect the newly created target record occurrence on all appropriate target sets. This is no simple task because:

1. at any given moment, all potential parent record occurrences may not exist,
2. a record occurrence may be incomplete because it may obtain partial data from each of several access paths.

During Linking, the Restructurer relies on user-specified primary key items for determining whether instances are unique. Each time a target record occurrence is created, the Restructurer must check all the records in the record type and compare keys with the newly created record. As may be obvious, this is not efficient, but is necessary given this particular implementation. Upon completion of this phase, control is returned to the Target Control phase.

Restructurer Implementation

The Restructurer was written primarily in ANSI FORTRAN to simplify coding and maximize program portability. In addition, some routines were coded in assembler language to perform specialized functions not easily accomplished in FORTRAN.

Approximately 5000 lines of FORTRAN code were required to implement the prototype Restructurer. It ran in 50K words on a Honeywell H6000, and required approximately 10,000 CPU seconds to construct a target RIF database containing 2500 record occurrences, an average of 4 seconds/record. Unfortunately, the Restructurer's execution time was found to be proportional to the square of the number of target record occurrences, and an average of 1500 seconds/record was projected for a target database of 150,000 record occurrences.

SUMMARY, PROBLEMS, AND CONCLUSIONS

Two approaches to the construction of restructuring languages and software for network databases have been presented. The elementary operations approach, although effective with hierarchical databases, does not extend well to network databases. The access path approach, in which all source structures are treated in the same way, is more naturally suited to complex network transformations, and powerful restructuring capabilities are obtained.

A Network Restructuring Language (NRL) based on the high-level specification of access paths was developed. It is essentially non-procedural, and its basic constructs—access paths and selection criteria—are familiar to users of network databases. These factors help to make it generally user-friendly.

The architecture of an operational NRL-driven Restructurer was discussed. The internal DBMS used as an implementation tool proved valuable in the construction of the Restructurer. It allows last-minute design changes to be incorporated easily and provides a facility for tuning internal data management.

Experience with the Restructurer has revealed two major problems. First, as noted above, execution efficiency was not a primary concern in its design. It was expected that the prototype Restructurer would be slow but practical to use, and that efficiency enhancements would be built into subsequent versions. Unfortunately, execution time was proportional to the square of the number of the target record instances. The core of the problem was the algorithm chosen for the Linking phase of restructuring, which led to a very large (approximately 3000 FORTRAN statements), very slow Linker module.

Secondly, although the construction of target record instances was easily specified in the NRL, the means by which target sets were established was quite opaque. In fact, it required a user writing NRL descriptions to understand certain features of the Restructurer algorithm. This difficulty compromised the NRL's claim to non-procedurality and user-friendliness.

Both of these problems have been corrected in a second version of the NRL and the Restructurer. The basic NRL structure remains unchanged although access paths are no longer required to be strictly linear—they may be tree-structured—and sets are explicitly established in a way that completely hides the restructuring algorithm from the user. Changes in the Restructurer's algorithm and enhancements to ADBMS, which maintains the source and target databases in their internal forms, have lowered execution time to a linear function of the size of the target database (with what appears to be an acceptable slope). The reader is referred to Reference 13 for theoretical details of the

second version, and to Reference 18 for usage-oriented details.

We feel justified in concluding that, using the access path approach, it is possible to build practical, general-purpose restructuring specification languages and restructuring software for network databases. However, since restructuring involves the exhaustive traversal of the source database—with some or all of the source data accessed many times—as well as the complete construction of a target database, it is an inherently slow process, and serious attention must be paid to a restructuring system's execution efficiency.

REFERENCES

1. Novak, D. and J. Fry, "The State of the Art of Logical Database Design," *Proc. Fifth Texas Conference on Computing Systems*, Austin, Texas, October 1976, pp. 30-39.
2. Fry, J. P., R. L. Frank, and E. A. Hershey, III "A Developmental Model for Translation," *Proc. 1973 ACM SIGFIDET Workshop on Data Description, Access and Control*, Denver, Colorado, November 1972, pp. 77-106.
3. Merten, A. G. and J. P. Fry, "A Data Description Approach to File Translation," *Proc. 1974 ACM SIGMOD Workshop on Data Description, Access and Control*, Ann Arbor, Michigan, May 1974, pp. 191-205.
4. Housel, B., V. Lum, and N. Shu, "Architecture to an Interactive Migration System (AIMS)," *Proc. 1974 ACM SIGFIDET Workshop on Data Description, Access and Control*, Ann Arbor, Michigan, May 1974, pp. 157-169.
5. Ramerez, J. A., N. A. Rin, and N. S. Prywes, "Automatic Conversion of Data Conversion Programs Using a Data Description Language," *Proc. 1974 ACM SIGFIDET Workshop on Data Description, Access and Control*, Ann Arbor, Michigan, May 1974, pp. 207-225.
6. Birss, E. W., and J. P. Fry, "Generalized Software for Translating Data," *Proc. 1976 NCC*, Vol. 45, AFIPS Press, Montvale, New Jersey, pp. 889-899.
7. Fry, J. P. and D. Jeris, "Towards a Formulation of Data Reorganization," *Proc. 1974 ACM SIGMOD Workshop on Data Description and Access*, New York, 1974.
8. Navathe, S. B. and J. P. Fry, "Restructuring for Large Data Bases," *ACM Transactions on Database Systems* 1, 2, June 1976, ACM, New York, 1976, pp. 138-158.
9. CODASYL DATA DESCRIPTION LANGUAGE COMMITTEE, *CODASYL Data Description Language Journal of Development*, June 1973, NBS Handbook 113, ACM, New York, January 1974.
10. Fry, J. P. and E. A. Sibley, "Evolution of Database Management Systems," *Computing Surveys*, 8, 1, March 1976, pp. 1-42.
11. Shoshani, A., "A Logical-Level Approach to Data Base Conversion," *Proc. ACM/SIGMOD International Conference on Management of Data*, ACM, New York, 1975.
12. Codd, E. F. and C. J. Date, "Interactive Support for Non-Programmers: The Relational and Network Approaches," *Data Models: Data-Structure-Set versus Relational*, R. Rustin (ed.) Ann Arbor, Michigan, May 1974, pp. 13-42.
13. Deppe, M. E., "A Relational Interface Model for Database Restructuring," Technical Report 76 DT 3, Data Translation Project, The University of Michigan, Ann Arbor, Michigan, 1975.
14. Deppe, M. E., and K. H. Lewis, "Data Translation Translation Definition Language Reference Manual for Version IIA Translator Release 1," Working Paper DT 5.2, Data Translation Project, The University of Michigan, Ann Arbor, Michigan, 1976.
15. Birss, E., M. Deppe, and J. Fry, "Research and Data Reorganization Capabilities for the Version IIA Data Translator," Data Translation Project Technical Report, February 1975, Graduate School of Business Administration, University of Michigan, Ann Arbor, Michigan.
16. Hershey, E. A. and P. W. Merslak, "A Data Base Management System for PSA Based on DBTG 71," ISDOS Working Paper No. 88, July 1975, ISDOS Research Project, Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, Michigan.
17. The Stored-Data Definition and Translation Task Group, "Stored-Data Description and Data Translation: A Model and Language," Information Systems.
18. Bodwin, James, et al., "The Translator Version IIA Release 2 User Manual," Technical [illegible] 76 DT 3.4, Data Translation Project, The University of Michigan, Ann Arbor, Michigan, forthcoming.
19. UNIVAC, UNIVAC [illegible] Data File Converter, Programmer Reference, UP-8070, Sperry [illegible] Corporation, March 1974.
20. Bakkom, D. E. and J. A. Eichyner, "Implementation of a Prototype Generalized File Translator," *Proc. 1975 ACM SIGMOD International Conference on Management of Data*, San Jose, California, May 1975, pp. 99-110.
21. Shoshani, A., "A Logical Level Approach to the Data Base Conversion," *Proc. 1975 ACM SIGMOD Conference*, San Jose, California, May 1975, pp. 112-122.
22. Shu, N. C., B. C. Housel, and V. Y. Lum, "CONVERT: A High-Level Translation Definition Language for Data Conversion," *Comm. ACM* 18, 10, October 1975, pp. 57-67.
23. Lewis, K., B. Driver, and M. Deppe, "A Translation Definition Language for Version II Translator," Working Paper 809, Data Translation Project, The University of Michigan, Ann Arbor, Michigan, 1975.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

THE INTELLIGENCE CYCLE - A DIFFERENTIATED PERSPECTIVE ON INFORMATION PROCESSING

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

16

The intelligence cycle—A differentiated perspective on information processing

by PETER G. W. KEEN
Stanford University
Stanford, California

ABSTRACT

This brief position paper presents a framework for mapping computer-based information aids onto the mental activities involved in the full problem-solving process. It argues that these activities are best described in terms of operators—the verbs and commands that the individual uses in a particular stage of the Intelligence Cycle.

The cycle begins with Discovery, the recognition of some signal requiring response. Discovery filters data into information and mainly involves operators that attenuate or amplify data: "alert," "keep track of" (amplification) and "summarize," "report averages" (attenuation). Few computer tools support amplification: The second stage, Interpretation is one where the machine generally outperforms the human mind, especially in inference and statistical analysis. Examples of operators for this stage are "compare," "review" and "suggest." The final stage, Analysis, is strongly supported by management science, especially through optimization models; typical operators are "test the impact of" and "evaluate."

None of our current tools supports the full Intelligence cycle. The position paper suggests that the development of a science of information-processing must both identify the cognitive operators underlying the activities within the cycle and match the technical building blocks to them.

INTRODUCTION

Developments in information-processing are mainly driven by technology; new tools generate new uses. As relational data base or pattern recognition methodologies move from the laboratory to commercial availability, we will extend our ability to help Knowledge Workers make more effective and more efficient use of their information resources. This process is fragmented and relies on serendipity; it also obscures the broader concern:

> What is the objective of the science of information-processing?
> What are the activities its tools support and augment?

The aim of this presentation is to step back and map the technology into its context—the *cycle* of Intelligence, a process of Discovery, Interpretation and Analysis[1] which begins from an initial awareness of a stimulus or problem and ends with a decision. By clarifying the operations within this cycle, we can both assess the techniques we now have and define the areas of most need and payoff. It is this analysis that should drive the technology and determine the tools we require.

The scheme presented here is a paradigm, a conceptual framework for organizing current knowledge, rather than a formal theory. It derives mainly from cognitive psychology, the science of human information-processing.[2] Its emphasis is descriptive; before we can prescribe tools for improving an activity, we must first describe its dynamics and context. In general, our technical focus has stressed prescription at the expense of this understanding.[3]

The paradigm clarifies some traditional distinctions in discussions of human response to and use of information:

structured	versus	unstructured
data	versus	information
qualitative	versus	quantitative
problem-finding	versus	problem-solving[4]

The presentation is influenced by Miller's[5] definition of a psychology for man-machine systems: as here, he similarly focuses on the *activity* of information-processing, in terms of operators—verbs such as "show me," "scan" and "summarize." Tools can best be understood in relation to the operators they support.[6]

THE CYCLE OF INTELLIGENCE

Most existing tools assist a Knowledge Worker who already has a purpose: a researcher scanning a database, an analyst reviewing a situation or a manager requesting summary reports. The Intelligence cycle begins well before this stage and includes Discovery—identifying and defining the problem and purpose. At an extreme, the Knowledge Worker sits daydreaming while the world around him randomly throws in his direction signal and noise. The cycle begins only when he is alerted to some signal.

Sometimes, the Discovery process is passive; he may not be actively scanning his environment. He may thus overlook "relevant" data.

Many commentators have emphasized that *data* becomes *information* by being filtered through some mental model that gives it meaning and relevance. The mind outperforms most computer-based tools in this process, which relies on alertness, pattern-creation and pattern-recognition. Of course, human limitations on memory, attention span and capacity for assimilating large masses of data make machine support invaluable in many instances.

Discovery filters data into information.[7] In many ways it is the key stage in the cycle, in that it involves problem-finding. Once alerted to a signal, we can generally respond to it if only through some barely adequate rule of thumb or standard operating procedure, but we may easily *overlook* a potentially critical signal. If we can scan more alertly or provide better filters, we reduce the risk of doing so, but the effort needed may exceed our resources. It is thus desirable to automate part of the filter. *What are the criteria for doing so?* The technical focus does not in itself define any criterion.

Stafford Beer,[8] the British cybernetician, describes the filter in terms of *attenuation* and *amplification*. Discovery reduces chaos to singularity. Huge volumes of data are attenuated through summary, selection and aggregation. Key signals are amplified and drawn to our attention. In general we have many tools for attenuation—for reducing the load on our limited attention and capacity; most reporting systems organize mass data into "meaningful" summary. Beer gives an example of amplification in the CYBERSTRIDE system he helped build for the Chilean government.[9] A forecasting algorithm, based on Bayesian decision theory, traps the stream of data on ongoing economic activity and tests if new signals imply a change in current trends and consequently a need to revise the mental model; if they do not *the data is ignored* and the Knowledge Worker not burdened with it. If the input implies a shift, the signal is amplified and the worker alerted to it.

The operators relevant to attenuation include "alert," "keep track of" and "locate any discrepancy." These are clearly different from "summarize" and "report averages," operators relating to attenuation. Information tools for Discovery thus need to be *differentiated*. We have far more tools for attenuation than for amplification; since the latter is central to problem-finding, this is clearly an area of great potential payoff in the selective development of new tools.

Attenuated information needs interpretation while amplification leads to Search, to a response to the signal and a more active effort at Discovery (see Figure 1). In either instance, only now is the worker aware of purpose and ready to select information and analytic aids. The second stage in the cycle, Interpretation, is largely inferential. The operators—the commands the worker gives to his tools—include "suggest," "review," "compare" and "deduce the (often statistical) meaning of." It is fairly easy to map techniques into this activity. Interactive Decision Support Systems such as Gerrity's portfolio management system are explicitly designed in terms of such operators as "SCAN" and "HISTO" (provide histograms).

While man outperforms machine in most aspects of Discovery, he is much less effective in Interpretation.[10] Tversky and Kahneman[11] emphasize the frequency with which individuals make simple statistical errors. Edwards[12] similarly points up our inability to make effective use of information we already have (to update probability estimates, for example). Tools such as MYCIN,[13] which aids medical diagnosticians in the process of inference, exploit the machine's comparitive advantage in Interpretation.

The final stage of the cycle is Analysis, the assessment of interpreted information; this usually results in some conscious decision. Discovery is mainly perceptual and therefore hard to observe or make explicit but Analysis is generally conscious, methodological and sequential.[14] It is concerned more with the use of information than information itself. Its operators include "evaluate," "compare these alternatives" and "test the impact of"—and of course "what if." The tools of management science—optimization and simulation models—obviously support these. It is not clear where man outperforms machine or vice versa. Many problem-solvers prefer to rely on their own intuitive methods although in structured situations they will rely on formal models.

Because Analysis is conscious and sequential, it is often constrained by time and computational effort. In many cases, we simplify the problem to the point where it is feasible for us to handle its demands, even if this involves misrepresentation—and sometimes peversion.[15] In many cases, by automating the operators it involves, we encourage the individual to make *more* comparisons, to enlarge his "bounded rationality;"[16] a frequent benefit cited for interactive computer systems is simply that they allow a user to test out more alternatives. More is not necessarily the same as better; such support may thus improve *efficiency* but not *effectiveness*.[17] In developing tools for Analysis, we must consider which operators they support and what "improvement" means (and is worth). Automative mechanical aspects of Analysis (comparisons, summarization) looses the bounds of rationality and *potentially* releases time for more attention to issues of effectiveness.

TOWARDS A DIFFERENTIATED PERSPECTIVE

The paradigm presented here can be expanded to give a fairly rigorous summary of both human information-processing activities and computer-based aids. The latter are the building blocks for a system to support the full cycle of Intelligence. If we focus on the operators needed by the Knowledge Worker, we will define those blocks in a far more differentiated way than we do now. The issue is not, simply for example, between qualitative and quantitative data or structured and unstructured problems. In the Discovery stage, attenuation generally involves numeric, and amplification judgmental, information. Our tools should reflect this.

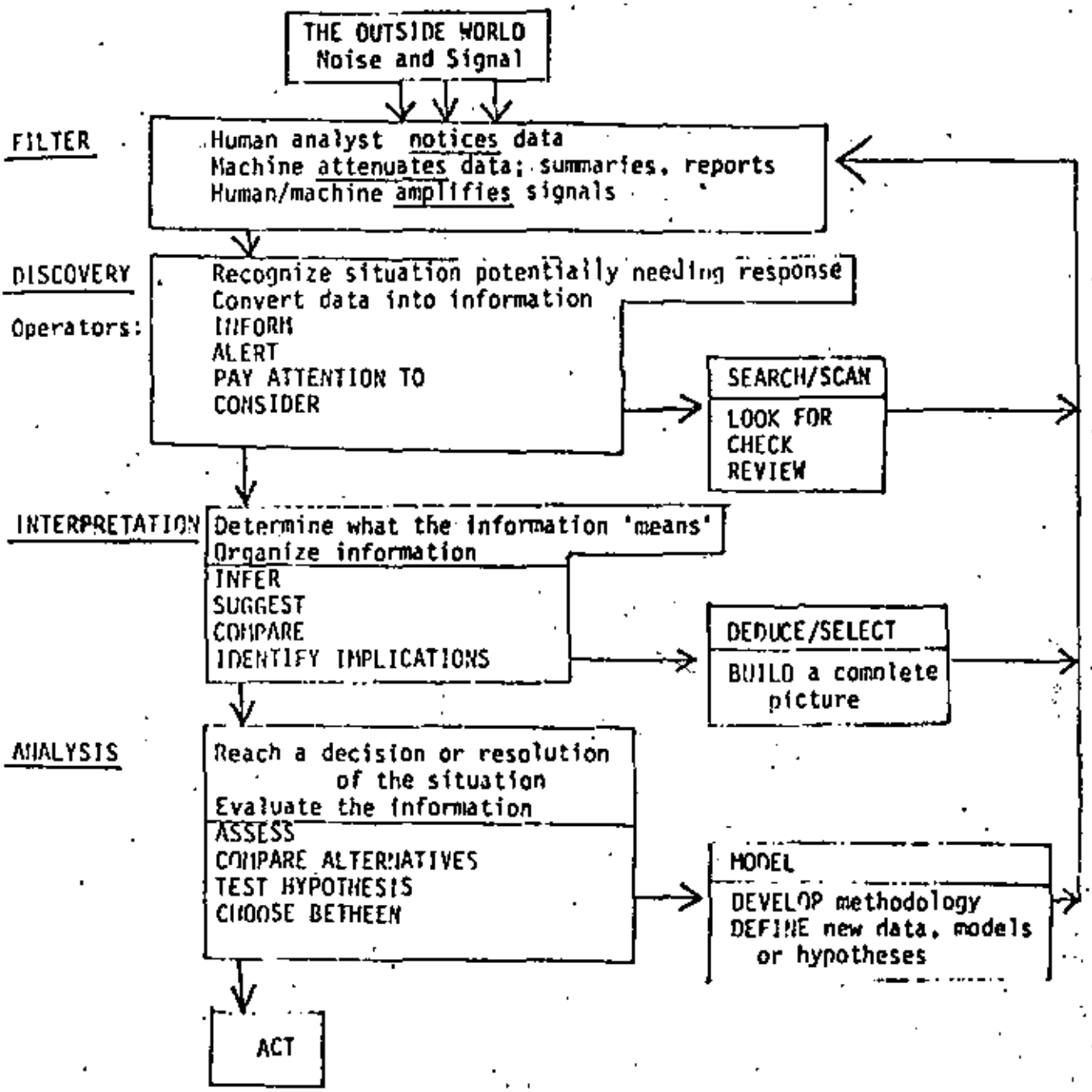


Figure 1—The cycle of intelligence

The paradigm implies selective development of new techniques, matched to specific operators, rather than imposition of the methods on the overall cycle. Any catalog of existing tools will show an abundance of aids for Interpretation and Analysis and near absence of support for Discovery. Any detailed examination of the operators involved in Discovery will, as a corollary, suggest that retrieval methods suitable for scanning and directed search are of little value for amplification and alerting the Knowledge Worker.

There is not space in this position paper to explore the paradigm in detail. The following assertions summarize its intent:

1. None of our tools can support the full Intelligence cycle, nor should we assume that they are other than building blocks;
2. In developing information aids, we must look at the activities involved in the cycle and draw on descriptive

and psychological models;
3. Tools are best defined in relation to operators;
4. We do not yet have a *science* of information processing; even if this paradigm is inaccurate or incomplete we need such frameworks that force us to develop a clearer sense of what our efforts should aim towards and what our techniques really are.

REFERENCES

1. cf Simon's model of the decision process: Intelligence, Design & Choice; Simon, H. A., *Elements of a Theory of Human Problem Solving*, Psychological Review Vol. 65, No. 3, May 1958.
2. More recently, the term "cognitive sciences" has been used by researchers to define a fusion between developmental psychology, Artificial Intelligence & information-processing theories.
3. Keen, P. G. W., and M. S. Scott Morton, *Decision Support Systems: An Organizational Perspective*, Addison-Wesley (in press), Chapter 3, Decision Making: Description versus Prescription.
4. This distinction is discussed in Leavitt, H. A., *Beyond the Analytic Manager*, California Management Review, Spring-Summer, 1975.
5. Miller, R. B., *Psychology for a Man-Machine Problem-Solving System*, Technical Report TR00 1246, IBM Data Systems Division, 1965.
6. Gerrity, T. P., *Design of Man-Machine Decision Systems: An Application to Portfolio Management*, Sloan Management Review, Winter 1971, for an example of a formal design strategy based on this approach.
7. Druzhinin, V. V., and D. S. Kontsevoi, *Concept, Algorithm and Decision*, Soviet Military Thought No. 6, U.S. Government Printing Office, 1972.
8. Beer, S., *Platform for Change*, Wiley, 1975.
9. Beer, S., "Fanfare for Effective Freedom," in Reference 8, pp. 423-451.
10. Dawes, R. M., "Objective Optimization under Multiple Subjective Functions," in J. L. Cochrane and M. Zeleny, eds., *Multiple Criteria Decision Making*, University of South Carolina Press, Columbia, South Carolina, 1973, 9-17.
11. Tversky, A., and D. Kahneman, *Judgment Under Uncertainty: Heuristics and Biases*, Science, Vol. 185, September 1974.
12. Edwards' work is summarized in Beach's comprehensive survey: Beach, B. H., *Expert Judgment About Uncertainty: Bayesian Decision Making in Realistic Settings*, Organizational Behavior and Human Performance, Vol. 14, pp. 10-59, 1975.
13. Shortliffe, E. H., S. G. Axline, B. G. Buchanan, and S. N. Cohen, "Design Considerations for a Program to Provide Consultations in Clinical Therapeutics," *Proceedings of the 13th San Diego Biomedical Symposium*, February, 1974.
14. However, there are significant differences between individuals' cognitive strategies and styles of problem-solving. See McKenney, J. L., and P. G. W. Keen, *How Managers' Minds Work*, Harvard Business Review, May 1974.
15. Taylor, R. N., *Psychological Determinants of Bounded Rationality and Implications for Decision Making Strategies*, Decision Sciences, Vol. 6, No. 5, pp. 409-429.
16. Simon, H. A., *Administrative Behavior*, (2nd Edition), McMillan, 1957.
17. See Reference 3, Chapter 1 for a discussion of the relevance of this distinction for the design of computer systems.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

IBM GENERAL PRODUCTS DIVISION PALO ALTO, CALIF.

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

17

W. Kent
IBM General Products Division
Palo Alto, California

ABSTRACT. The simplifying assumptions often made about entities and relationships in data structures don't always hold in real life.

INTRODUCTION

A message to mapmakers: highways are not painted red, rivers don't have county lines running down the middle, and you can't see contour lines on a mountain.

For some time now my work has concerned the representation of information in computers. The work has involved such things as file organizations, indexes, hierarchical structures, network structures, relational models, and so on. After a while it dawned on me that these are all just maps, being poor artificial approximations of some real underlying terrain.

These structures give us useful ways to deal with information, but they don't always fit naturally, and sometimes not at all. Like different kinds of maps, each kind of structure has its strengths and weaknesses, serving different purposes, and appealing to different people in different situations. Data structures are artificial formalisms. They differ from information in the same sense that grammars don't describe the language we really use, and formal logical systems don't describe the way we think. "The map is not the territory" [Hayakawa].

What is that territory really like? How can I describe it to you? Any description I give you is just another map. But we do need some language (and I mean natural language) in order to discuss this subject, and to articulate concepts. Such constructs as "entities", "categories", "names", "relationships", and "attributes" seem to be useful. They give us at least one way to organize our perceptions and discussions of information. But we need to have in mind an understanding of the real nature of such constructs -- especially the difficulties involved in trying to define or apply them precisely. We should appreciate that such terminology does not report objective states of reality; instead, these terms record our subjective perceptions. And such perceptions are different for different people and different purposes, and they can vary with time.

We can make simplifying assumptions when we design data structures and systems. But there's not much use in fooling ourselves about the nature of information. It leads to wasted effort (in seeking "correct" descriptions), and needless fault-finding ("you described it incorrectly!").

Of the various constructs applicable to the description of information, this paper explores the nature of two: entities and relationships.

ENTITIES

> "Entities are a state of mind. No two people agree on what the real world view is." [Metaxides]

An information system (e.g., data base) is a model of a small, finite subset of the real world. We expect certain correspondences between constructs inside the information system and in the real world. We expect to have one record in the employee file for each person employed by the company. If an employee works in a certain department, we expect to find that department's number in that employee's record.

So, one of the first concepts we have is a correspondence between things inside the information system and things in the real world. Ideally, this would be a one-to-one correspondence, i.e., we could identify a single construct in the information system which represented a single thing in the real world.

We're in trouble already. In the first place, it's not so easy to pin down what construct in the information system will do the representing. It might be a record (whatever that means), or a part of one, or several of them, or a catalog entry, or a subject in a data dictionary, or .... For now let's just call it a "representative", and come back to that topic later. The other half of the problem concerns the things we want represented in the system.

As a schoolteacher might say, before we start writing data descriptions let's pause a minute and get our thoughts in order. Before we go charging off to design or use a data structure, let's think about the information we want to represent. Do we have a very clear idea of what that information is like? Do we have a good grasp of the semantic problems involved?

Becoming an expert in data structures is like becoming an expert in sentence structure and grammar; it's not of much value if the thoughts you want to express are all muddled.

The information in the system is part of a communication process among people. There is a flow of ideas from mind to mind; there are translations along the way, from concept to natural languages to formal languages (constructs in the machine system) and back again. An observer of (or participant in) a certain process recognizes that a certain person has become employed by a certain department. The observer causes that fact to be recorded; perhaps in a data base, where someone else can later interrogate that recorded fact to get certain ideas out of it. The resemblance between the extracted ideas and the ideas in the original observer's mind depends not only on the accuracy with which the messages are recorded and transmitted. It also depends heavily on the participants' common understanding of the elementary references to "a certain person", "a certain department", and "is employed by".

[illegible]

What is "one thing"?

That appears at first to be a trivial, irrelevant, irreverent, absurd question. It's not. The question illustrates how deeply ambiguity and misunderstanding are ingrained in the way we think and talk.

Consider those good old workhorse data base examples, parts and warehouses. We normally assume a context in which each part has a part number and occurs in various quantities at various warehouses. Notice that: various quantities of one thing. Is it one or many? Obviously, the assumption here is that "part" means one kind of part, of which there may be many physical instances. (The same ambiguity shows up very often in natural usage, when we refer to two physical things as "the same thing" when we mean "the same kind".) It is a perfectly valid and useful point of view in the context of, e.g., an inventory file: we have one representative (record) for each kind of thing, and speak loosely of all occurrences of the thing as collectively being one thing. (We could also approach this by saying that the representative is not meant to correspond to any physical object, but to the abstracted idea of one kind of object. Nonetheless, we do use the term "part", and not "kind of part".)

Now consider another application, a quality control application, also dealing with parts. In this context, "part" means one physical object; each part is subjected to certain tests, and the test data is maintained in a data base separately for each part. There is now one representative in the information system for each physical object, many of which may have the same part number.

In order to integrate the data bases for the inventory and quality control applications, the people involved need to recognize that there are two different notions of "thing" associated with the concept of "part", and the two views must be reconciled. They will have to work out a convention wherein the information system can deal with two kinds of representatives: one standing for a kind of part, another standing for one physical object.

I hope you're convinced now that we have to go to this depth to deal with the basic semantic problems of data description.

We are dealing with a natural ambiguity of words, which we as human beings resolve in a largely automatic and unconscious way, because we understand the context in which the words are being used. When a data file exists to serve just one application, there is in effect just one context, and users implicitly understand that context; they automatically resolve ambiguities by interpreting words as appropriate for that context. But when files get integrated into a data base serving multiple applications, that ambiguity-resolving mechanism is lost. The assumptions appropriate to the context of one application may not fit the contexts of other applications.

There are a few basic concepts we have to deal with here:

- Oneness.
- Sameness. When do we say two things are the same, or the same thing? How does change impact identity?
- What is it? In what categories do we perceive the thing to be? What categories do we acknowledge? How well defined are they?

These concepts and questions are tightly intertwined with one another.

Consider "book": If an author has written two books, a bibliographic data base will have two representatives. If a lending library has five circulating copies of each, it will have ten representatives in its files. After we recognize the ambiguity we try to carefully adopt a convention using the words "book" and "copy". But it is not natural usage. Would you understand the question "How many copies are there in the library?" when I really want to know how many physical books the library has altogether?

There are other connotations of the word "book" that could interfere with the smooth integration of data bases. A "book" may denote something with hard covers, as distinguished from things in soft covers like manuals, periodicals, etc. Thus a manual may be classified as a "book" in one library but not in another. I don't always know whether conference proceedings comprise a "book".

A "book" may denote something bound together as one physical unit. Thus a single long novel may be printed in two physical parts. (When we recognize the ambiguity, we sometimes try to avoid it by agreeing to use the term "volume" in a certain way, but we are not always consistent. Sometimes several "volumes" are bound into one physical "book".) We now have as plausible perceptions: the one book written by an author, the two books in the library's title files (Vol. I and Vol. II), and the ten books on the shelf of the library which has five copies of everything.

Incidentally, the converse sometimes also happens, as when several novels are published as one physical book (e.g., collected works).

So, once again, if we are going to have a data base about books, before we can know what one representative stands for, we had better have a consensus among all users as to what "one book" is.

Going back now to parts and warehouses, the notion of "warehouse" opens up another kind of ambiguity. There is no natural, intrinsic notion of what comprises "one warehouse". It may be a single building, or a group of buildings separated by any arbitrary distance. Several warehouses (e.g., belonging to different companies) may occupy the same building (perhaps on different floors). So, what is "one warehouse"? Anything that a certain group of people agrees to call a warehouse. Given two buildings, they might agree to treat them as one, two, or any number of warehouses -- with all perceptions being equally "correct".

Thus, the boundaries and extent of "one thing" can be very arbitrarily established. This is even more so when we perform "classification" in an area that has no natural sharp boundaries at all. The set of things that human beings know how to do is infinitely varied, and changes from one human being to another in the most subtle and devious ways. Nonetheless, the "skills" portion of a personnel data

base asserts a finite number of arbitrary skill categories, with each skill being treated as one discrete thing, i.e., it has one representative. The number and nature of these skills is very arbitrary (i.e., they do not correspond to natural, intrinsic boundaries in the real world), and they are likely to be different in different data bases. Thus, a "thing" here is a very arbitrary segment partitioned out of a continuum. This applies also to the set of subjects in a library file or information retrieval system, to the set of diseases in a medical data base, to colors, etc.

This classification problem underlies the general ambiguity of words. The set of concepts we try to communicate about is infinite (and non-denumerable in the most mind-boggling sense), whereas we communicate using an essentially finite set of words. (For this discussion, it suffices just to think about nouns.) Thus, a word does not correspond to a single concept, but to a cluster of more or less related concepts. Very often, the use of a word to denote two different ideas in this cluster can get us into trouble.

A case in point is the word "well" as used in the data files of an oil company. In their geological data base, a "well" is a single hole drilled in the surface of the earth, whether or not it produces oil. In the production data base, a "well" is one or more holes covered by one piece of equipment, which has tapped into a pool of oil. The oil company is currently having trouble integrating these data bases to support a new application: the correlation of well productivity with geological characteristics.

HOW MANY THINGS IS IT?

A single physical unit often functions in several roles, each of which is to be represented as a separate thing in the information system. Consider a data base maintaining statistics on defensive plays by a baseball team, both on a position basis and on an individual basis. The data base might have representatives for 34 things: nine positions and 25 players. When Joe Smith, playing second base, makes a putout, the data about two things is modified: the number of putouts by Joe Smith, and the number of putouts by a second baseman. That human figure standing near second base is represented as (and is) two things: Joe Smith and the second baseman.

Consider the question of "sameness". Suppose Joe switches to shortstop, and makes another putout. Did the same thing make these two putouts? Yes - Joe Smith made both. No - one was made by the second baseman, the other by the shortstop.

Why is that human figure perceived and treated as two things, rather than one or three or ninety-eight? Not by any natural law, but by the arbitrary decision of some human beings, because the perception was useful to them, and corresponded to the kinds of information they were interested in maintaining in the system.

If the file only had data about player positions, then the same physical object would be treated as being different things at different times. Joe is sometimes a second baseman and sometimes a shortstop. This particular file doesn't even know that Joe exists, and can play both positions.

Also consider the related people (e.g., husband and wife, father and son) who work for the same company. When considering medical benefits, each of these people has to be considered twice: once as an employee, and once as a dependent of an employee.

Or suppose a person held two jobs with the company, on two different shifts. Does that signify one or two employees? Shipping clerk John Jones and third-shift computer operator John Jones might be the same person. Does it matter? Sometimes.

It is plausible (bizarre, perhaps, but plausible) to view a certain employee and a certain stockholder as two different things, between which there happens to exist the relationship that they are embodied in the same person. There would then exist two representatives in the system, one for the employee and one for the stockholder. It's perfectly all right, so long as users understand the implications of this convention (e.g., deleting one might not delete the other).

Transportation schedules and vehicles offer other examples of ambiguities. In the use of such terms as "flight" and "plane" (even if we ignore the other definitions of "plane" having nothing to do with flying machines). What does "catching the same plane every Friday" really mean? It may or may not be the same physical airplane. But if a mechanic is scheduled to service the same plane every Friday, it had better be the same physical airplane. And another thing: if two passengers board a plane together in San Francisco, with one holding a ticket to New York and the other a ticket to Amsterdam, are they on the same flight?

Classification, e.g., of skills, impacts the notion of "sameness" as much as the notion of "how many". The way we partition skills determines both how many different things we recognize in this category, and when we will judge two things to be the same. Consider a group of people who know how to do such things as paint signs on doors, paint portraits, paint houses, draw building blueprints, draw wiring diagrams, etc. One classifier might judge that there is just one skill represented by all of these capabilities, namely "artist", and that every person in this group had the same skill. Another classifier might claim there are two skills here, namely painting and drawing. Then the sign painter has the same skill as the portrait painter, but not the blueprint drawer. And so on.

The same game can be played with colors. Two red things are the same color. What if one is crimson and the other scarlet?

The perceptive reader will have noticed that two kinds of "how many" questions have been intermixed in this section. At first we were exploring how many kinds of things something might be perceived to be. But occasionally we were trying to determine whether we were dealing with one or several things of a given kind. (If you can't apply that distinction to the preceding discussions, then please don't become a data base administrator. I fear your data base may well become a minefield of semantic traps.)

CHANGE

And then there's change. Even after consensus has been reached on what things are to be represented in the information system, the

impact of change must be considered. How much change can something undergo and still be the "same thing"? At what point is it appropriate to introduce a new representative, because change has transformed something into a new and different thing?

The problem is one of identifying or discovering some essential invariant characteristic of a thing, which gives it its identity. That invariant characteristic is often hard to identify, or may not exist at all.

We seem to have little difficulty with the concept of "one person" despite changes in appearance, personality, capabilities, and, above all, chemical composition (the proportions and structure -- i.e., the chemical formulas -- may not change much, but the individual atoms and molecules are continually being replaced... again illustrating an ambiguity between "same kind" and "same instance"). When we speak of the same person over a period of time, we certainly are not referring to the same ensemble of atoms and molecules. What then is the "same person"? We can only appeal to some vague intuition about the "continuity" of -- something -- through gradual change. The concept of "same person" is so familiar and obvious that it is absolutely irritating not to be able to define it. (Definitions in terms of "soul" and "spirit" may be the only true and humanistic concepts, but, significantly, we don't know how to deal with them in a computer-based information system.) It is only when the notion of "person" is pushed to some limit do we realize how imprecise the notion is. This is the basis of some legal issues.

Modern medicine is dissecting our concept of "person" via transplanted and artificial limbs and organs. At one time people believed the heart to be the seat of the emotions; they might have argued that the recipient of a heart transplant became the person who the donor was -- he had surely taken up residence in a new body. We are more likely to take that position with respect to the brain, rather than the heart. A number of legal issues will have to be resolved when brain transplants begin to be performed (and the issues may get more complex if just portions of the brain are transplanted).

In an information system maintaining data about people, we will have to decide which information goes interchanged between two representatives. Which information is to be associated with the body, and which with the brain? A name? A spouse? Other relatives? How is the medical history rearranged? Who has which job? Skills? Financial obligations?

Similar questions apply to organizations, such as companies, departments, teams, government agencies, etc. Is it still the same company after changes in employees? (Of course.) Management? (Yes.) Owners? (Maybe.) Buildings and facilities? (Yes.) Locations? (Probably.) Name? (Probably). Principal business? (Maybe.) State and country of incorporation? (Maybe.) The answers are significant to the handling of old contracts and other obligations, the determination of employee vacation and retirement benefits, etc.

And political boundaries. A data base of population statistics must have some definition of what is meant by India, Pakistan, Germany, Czechoslovakia, etc., over time. (There's more involved than a change of name; the things themselves have been created, destroyed, merged, split, re-partitioned, etc.). In some other data base it

may have to be understood that two people born at different times in the same town might have been born in different countries.

A whole spectrum of concepts. There is the "one 'book" containing the ideas expressed by an author, which is the same book regardless of which language it is translated into, or how it is edited, abridged, condensed, revised, etc.

Then there are "editions", which differ from each other by some arbitrary amount, due either to changes in the content or to the correction of significant amounts of error. On the other hand, some minor amount of difference (erroneous or deliberate) is permitted between reprints of a single edition.

A condensation or abridgement may be grossly different from the original, but for some purposes it is treated as being the same book.

This topic is most painfully familiar to us in relation to "versions", e.g., of such things as programs. There is some arbitrary threshold up to which minor changes can be made without creating a new version. The old copy is discarded, there may or may not be a record of the modification, and the representative (e.g., catalog entry) of the old copy now serves to represent the new copy.

Beyond a certain (arbitrary) point, we decide to keep the old and new copies as different versions. We now enter a metaphysical realm in which we manage to merge the concepts of "one" and "many", as in the expressions "these several things are different versions of the same thing". In some contexts we mean to refer to all versions collectively (as in the property: this is a Fortran compiler), in some we refer to a particular copy, and in some we refer to one copy -- whichever one happens to be the "current" version.

WHAT IS IT?

We have so far been focussing on the questions of "oneness" and "sameness". That is, given that you and I are pointing to some common point in space (or we think we are), and we both perceive something occupying that space (perhaps a human figure), how many "things" should that be treated as in the information system? One? Many? Part of a larger thing? Or not a thing at all?

And: do we really agree on the composition and boundary of the thing? Maybe you were pointing at a brick, and I was pointing at a wall.

And: if we point to that same point in space tomorrow (or think we are), will we agree on whether or not we are pointing at the same thing as we did today?

Some of this focusses on what the thing *is*. I don't mean its properties, like is it solid, or is it red, or how much does it weigh, but what is it? I had to use the phrase "human figure" above because I didn't think you would follow my point if I kept using the indefinite word "thing" -- I had to convey some kind of tangible example. But that phrase is just one possible perception of the "thing" we pointed to. You might have said it was a mammal, or a man, or a solid object, or a bus driver, or your father, or a stockholder, or a customer, or ... ad nauseum.

I will refer to what a thing is -- or at least what it is described to be in the information system -- as its "category" (agreeing with the usage in, e.g., [Abrial]). Like everything else, the treatment of categories requires a number of arbitrary decisions to be made.

There is no natural set of categories. The set of categories to be maintained in an information system must be specified for that system. In one system it might be employees and customers, in another it might be employees and dependents, or enrolled computer users, or plaintiffs and defendants, and in an integrated data base it might include all of these. A given thing (representative) might belong to many such categories.

Not only are there different kinds of categories, but categories may be defined at different levels of refinement. One application might perceive savings accounts and loan accounts as two categories, while another perceives the single category of accounts, with "savings" or "loan" being a property of each account. In another case, we might have applications dealing with furniture or trucks or machines, while another deals with capital equipment (assigning everything a unique inventory number). Thus, some categories are, by definition, subsets of others, making a member of one category automatically a member of another. Some categories overlap without being subsets. For example, the category of customers (or of plaintiffs, in a legal data base), might include some people, some corporations or other businesses, and some government agencies.

It is often a matter of choice whether a piece of information is to be treated as a category, an attribute, or a relationship. This corresponds to the equivalence between "that is a parent" (the entities are parents), "that person has children" (the entities are people, with the attribute of having children), and "that person is the parent of those children" (the entities are people and children, related by parentage).

It's often difficult to determine whether or not a thing belongs in a certain category. Almost all non-trivial categories have fuzzy boundaries. That is, we can almost always think of some object whose membership in the category is debatable. The usual treatment of this situation is either to leave it to the arbitrary decision of whoever is categorizing the object, or to establish a locally understood set of rules which probably don't match the rules used for the same category in somebody else's information system. Just as an example, consider the simple and "well understood" category of "employee". Does it include part-time employees? Contract employees? Employees of subsidiary companies? Former employees? Retired employees? Employees on leave? On military leave? Someone who has just accepted an offer? Signed a contract but not yet reported for work? Not only do the answers have to be decided according to how the company wants to treat the data, but perhaps the questions can't even be answered consistently within the company. A person on leave may not be an employee for payroll purposes, although he is for benefits purposes. Then the notions of category and property have to be reexamined again, to arrive at a set meaningful to all users.

As another example, consider the category of "cars", and decide if the following are included: station wagons, micro-buses, ordinary buses, pickup trucks, ordinary trucks, motor homes, dune buggies,

racing carts, motorcycles, ... What about a home-made contrivance in which a short pickup truck bed is hung out of the trunk of a sedan? An old bus converted to a motor home?

"A more amusing example is to imagine a continuum of physical objects between some given chair and table, constructed by letting the chair back shrink while its seat expands and flattens, and its legs become higher. There will be some strange objects in this continuum which cannot clearly be assigned to either class" (Gegwen).

The editor of a collection is generally listed as the "author" of the book. Did he "author" anything?

The category of a thing (i.e., what it is) might be determined by its position, or environment, or use, rather than by its intrinsic form and composition. In the set of plastic letters my son plays with, there is an object which might be an "M" or a "W", depending on how he holds it. Another one could be a "u" or an "n", and still another might be "b", "p", "d", or "q". I never know whether to tell him that the "I" is a "i", "j", or "l".

Does the distinction between a bench and a table depend on your height?

I can imagine the same hollow metal tube being called a pipe, an axle, a lamp pole, a clothes rack, a mop handle, a shower curtain rod, and how many more can you name? A nail driven into a wall might be designated a coat hook.

In part, these observations illustrate the difficulty of distinguishing between the category (essence) of a thing and the uses to which it may be put (its roles).

There are also interesting questions having to do with fragments of things, and imitations. Is it still a donut after you've taken a bite out of it? Did you ever call a stuffed toy an animal?

The fundamental problem -- this paper -- is self describing. Just as it is difficult to precisely partition a subject like personnel data into neat categories, so also is it difficult to precisely partition a subject like "information" into neat categories like "categories", "entities", and "relationships". Nevertheless, in both cases, it's much harder to deal with the subject if we don't attempt some such partitioning.

RELATIONSHIPS

A relationship is an association among several things, with that association having a particular significance. For brevity, I will refer to the significance of an association as its "reason". There is an association between you and your car, for the reason that you own it. There's an association between a teacher and a class, because he teaches it. There's an association between a part and a warehouse, because the part is stored there. Much of the data in an information system is about such relationships.

Relationships can be named, and for now we will treat the name as being a statement of the reason for the association (which means we still sometimes invent names which are whole phrases, such as "is-employed-by"). As usual, we have to be careful to avoid confusion between kinds and instances. We often say that "owns" is a relationship, but it is really a kind of relationship of which there are many instances: your ownership of your car, your ownership of your pencil, someone else's ownership of his car. For now, we will generally use the unqualified term "relationship" to mean a kind, and add the term "instance" if that's what is meant. (So, to be precise, our opening definition was of a relationship instance. A relationship then becomes a collection of such associations having the same reason.)

Note that the reason is an important part of the relationship. Just identifying the pair of objects involved is not enough; several different relationships can exist among the same objects. Thus, if the same person is your brother, your manager, and your teacher, these are instances of three different relationships between you and him.

DEGREE, DOMAIN, AND ROLE

We have so far looked only at relationship instances involving two things. They can also be of higher "degree". If a certain supplier ships a certain part to a certain warehouse, then that is an instance of a relationship of degree three. If that supplier uses a certain trucking company to ship that part to that warehouse, then we have a fourth degree relationship.

We must distinguish between "degree" and a confusingly similar notion. If a department employs four people, we might view that as an association among five things. If another department employs two people, we have an association among three things, and we couldn't say in general that the "employs" relationship has any particular degree.

We proceed out of this dilemma in several steps. As a first approximation, think of a relationship (not an instance) as a pattern, given as a sequence of categories (e.g., departments and employees). An instance of such a relationship then includes one thing from each category (i.e., one department and some employees). The degree of such a relationship would then be the number of categories in the defining pattern. What we have done is to reduce the "employs" relationship from being an association between one department and all of its employees to being an association between one department and one of its employees. Although the former is certainly a legitimate relationship, it is difficult to subject it to any definitional discipline. We will only deal with relationships in the latter form.

Specifying the pattern of a relationship as a sequence of categories is sometimes too restrictive. There are many relationships which permit several categories to occur at the same "position", as is the case when one can "own" many kinds of things. We therefore introduce the term "domain" to designate all the things which may occur at a given position in the relationship. A domain may include several categories. Thus we might describe an "owns" relationship as having two domains, with the first domain including such catego-

ries as employees, departments, and divisions, while the second domain included such categories as furniture, vehicles, stationery supplies, computers, etc.

"Domain" and "category" could be treated as the same concept if (1) we are dealing with a system which permits overlapping categories, e.g., unions and subsets; (2) the system does not impose intolerable performance or storage penalties for maintaining many declared categories; and (3) it doesn't bother our intuitions to think of all owners of things as a single kind of entity, and all owned things as another single kind.

One final improvement in the specification of relationships makes the specification more informative and less formally structured. Instead of assigning a domain to a sequential position in a pattern, we can give it a unique "role" name describing its function in the relationship, such as "owner" and "owned". Thus a relationship can be specified as an unordered set (rather than a sequential pattern) of unique role names. The number of role names is the degree of the relationship. A domain is specified for each role.

Role names are especially useful when several roles draw from the same domain. A "manages" relationship would be defined over the roles "manager" and "managed", both drawing from the domain of employees.

FORMS OF BINARY RELATIONSHIPS

Much of the information in an information system is about relationships. However, most data models (e.g., the relational model, IMS hierarchies, DBTG networks) do not provide a direct way to describe such relationships, but provide instead a variety of representational techniques (record formats, data structures). Implicit in most of these, and in the accompanying restrictions in the data processing system, is the ability to support some forms of relationships very well, some rather clumsily, and some not at all.

In order to assess the capabilities of a data model, it would help to have some systematic understanding of the various forms of relationships which can occur in real information. In the next few paragraphs I will discuss some significant characteristics of relationships. A particular "form" of a relationship is then some combination of these characteristics. A method for assessing a data model would include a determination of which forms it supported well, poorly, or not at all. Note the emphasis on combinations. In most data models you can probably manage to find a way to obtain most of the following features, taken one at a time. The challenge is to support relationships having various combinations of these features.

By "support", I mean that

- the system somehow permits a constraint to be asserted for the relationship (e.g., that it is one-to-many), and
- the system thereafter enforces the constraint (e.g., will not record an employee's assignment to more than one department at a time).

Such support is often implicit in the data structure (e.g., hierarchy), rather than being declared explicitly.

The set of characteristics listed below is probably incomplete; I imagine it will always be possible to think of additional relevant criteria. For simplicity, we are now only considering "binary" relationships, i.e., those of degree two (two roles). The concepts can be readily generalized to "n-ary" relationships (those of any degree).

Complexity

A relationship might be one-to-one (departments and managers; husbands and wives), one-to-many (departments and employees), or many-to-many (students and classes; parts and warehouses; parts assemblies). The relationship between employees and their current departments is (typically) one to many, whereas the relationship between employees and all the departments they have worked in (as recorded in personnel files) is many to many.

Another way to characterize complexity is to describe each direction of the relationship separately as simple (mapping one element to one) or complex (mapping one element to many). (The terms "singular" and "multiple" are also used.) Thus "manager of department" is simple in both directions; "manager of employee" is simple in one direction and complex in the other. Relative to the number of "forms" of relationships, this would count as four possibilities, since a given relationship might be simple or complex in each direction.

One advantage to this latter view is that it corresponds well with certain aspects of data extraction. Very often a relationship is being traversed in one direction (e.g., find the department of a given employee); the data processing system usually has to anticipate whether the result will contain one element or many (e.g., whether an employee might be in more than one department). The complexity of the reverse direction is of little concern (i.e., whether or not there are also other employees in the department).

Thus, if a given direction is complex, it doesn't matter much whether the relationship is 1:n or m:n. If the direction is simple, the distinction between n:1 and 1:1 may be immaterial.

Category Constraints

Either side of a binary relation might be constrained to a single

category, constrained to any of several specified categories, or unconstrained (three possibilities on each side, for a total of nine combinations). Constraint to a single category is probably the most common situation, as in the examples above under "Complexity".

Constraint to a set of categories occurs, for example, when a person can "own" things in several different categories, or when the owner might be a person, department, division, company, agency, or school. This case might be avoided by defining one new category as the union of the others -- if you're dealing with a data model which permits overlapping categories.

It is hard to think of a relationship which is naturally unconstrained as to category (i.e., one that applies to every kind of thing), but it often makes sense to handle a relationship that way in a real data processing system. Perhaps the relationship does happen to apply to all of the things represented in this particular data base, or to so many of them that it isn't worth checking for the few exceptions. Perhaps the installation doesn't want to incur the overhead of enforcing the constraint, and trusts the applications to assert only sensible relationships. Or, the system simply may not provide any mechanism for asserting and enforcing such constraints.

Self-Relation

Three possibilities:

- the relationship is not meaningful between things in the same category;
- things in the same category may be so related, but a thing may not be related to itself;
- things may be related to themselves.

The first case is again probably the most common. The second occurs, for example, in organization charts and parts assemblies. Examples of the third are our representatives in government (the representative is one of his own constituents), and canvassers for fund drives (the canvasser collects from himself).

Incidentally, I am thinking here of the simple case where categories are mutually exclusive. When categories overlap, as in subsets, things may be more complicated.

Optionality

On either side of the binary relationship, the relationship might be optional (not everybody is married) or mandatory (every employee must have a department). I will count this as four combinations (two possibilities on each side), although there could conceivably be more; one of the domains may include several categories, with the relationship being optional in some categories and mandatory in others.

The Number of Forms

Even with this limited list of characteristics, we already have 432 forms (4 x 9 x 3 x 4). This number might include some symmetries, duplicates, and meaningless combinations, but after subtracting these we still have a sizeable checklist.

Multiplicity of Relationships

Another important criterion concerns whether and how the system permits more than one relationship over the same pair of domains. The nature of the support often varies according to the forms of the relationships involved.

Examples

A sampling of a few forms and how they are handled in some data models.

For instance: certain cases can only be implemented in IMS using logical relationships (intersection records), e.g., self-relation, or multiple relationships over the same domains. These cases cannot also be constrained to be one-to-many relationships, since they are no longer part of a single hierarchic structure.

Consequently, that most elementary of structures, the homogeneous hierarchy (like an organization chart), cannot be represented in either IMS or DBTG.

Also, there is no way to enforce a one-to-one relationship in IMS, except by representing both entities within the same segment. Then it becomes difficult to change the relationship, or to make it optional -- and you can't have an application that looks at one entity without the other.

Some systems (e.g., [IMS], [DBTG]) require 1:n relationships to have only a single category on the "parent" or "owner" side of the relationship. Consider a 1:n relationship which naturally has parents in several categories (e.g., suppose that items of capital equipment may be owned by departments or divisions, but not both). It is sometimes suggested that such a relationship can be modelled as the composition of several 1:n relationships, one for each parent category (e.g., one relationship for things owned by departments, and another for things owned by divisions). This doesn't usually work, however, because it is difficult to prevent an item from having a parent in each of these relationships (i.e., that data structure would erroneously permit a piece of equipment to be owned by a department and a division at the same time). Furthermore, this approach creates an unnatural semantics by replacing one natural relationship by two artificial ones. One can no longer ask "Who owns this equipment?" One now has to engage in a stilted dialogue: "Which department owns this equipment? None? Oh, then which division owns it?"

RELATIONSHIPS AND INSTANCES ARE ENTITIES

Instances of relationships are things themselves, about which we may have information in the system.

They have attributes. Just as you have an age, so does your association with your department, your spouse, and your car. For a given part (type) stocked in a given warehouse, there is a certain quantity on hand.

They can be related to other things. A certain part stocked in one warehouse is to be re-ordered from one supplier; the same part stocked in a different warehouse may be re-ordered from a different supplier.

Instances of relationships can be related to each other (illustrated in the next section).

Relationships have names, and could be subject to the general variety of naming conventions. The names of relationships comprise valid information. An information system should be able to answer questions like:

* What relationships exist between x and y?
* In what relationships is x involved?

Instances of a relationship can be identified (named) by identifying the relationship and the entities being related: "employed-in, John Jones, Accounting". In real systems these composite names of instances are usually not represented explicitly, but are implied by the definition and organization of the records in a file. A record in an employee file will explicitly contain "John Jones" and "Accounting"; "employed-in" is implicitly understood by users of the file, or may be factored out into a file or record description somewhere. (Are there real examples where an instance of a relationship has a single simple name of its own, rather than a composite name?)

A relationship (not an instance) is itself a represented entity, since we have definitional information about it in the system ("data about data").

BINARY AND N-ARY RELATIONSHIPS

An "n-ary" relationship has degree n (n roles). "Binary" and "ternary" relationships have degree two and three, respectively.

Relationships of degree greater than two can be described as combinations of binary relationships. This follows from the observation that an instance of a relationship is itself a thing which can be related to other things. Consider a ternary relationship among parts, warehouses, and suppliers, wherein a given part may be ordered from certain suppliers for one warehouse and from different suppliers for another warehouse.

We can start with the binary relationship between parts and warehouses, i.e., which part is stored in which warehouse. We can call this relationship "allocations", and we can call each instance of a part assigned to a warehouse an "allocation". [Notice the influence of language on our thinking. By having the single noun "allocation" for it, we can comfortably think of the instance of the relationship as being a thing in itself. Calling it "a part stored in a warehouse" doesn't give it the feeling of a "thing".]

We next define the binary relationship between suppliers and allocations. This binary relationship literally identifies which supplier services which allocation. But, through our understanding of allocations, we know that it really means which supplier sends which part to which warehouse. We have thus defined the ternary relationship as a pair of binary relationships, which we can denote symbolically as PW and S(PW).

[Notation: "PW" stands for the binary relationship between parts and warehouses. "SPW" stands for the ternary relationship between suppliers, parts, and warehouses. "S(PW)" stands for a particular decomposition of this relationship; it is the binary relationship between suppliers and the binary relationship between parts and warehouses -- i.e., it is the binary relationship between suppliers and allocations. For simplicity, we are assuming that one domain is the same as one category. Also, the ordering of domains is not relevant to this discussion, and we can therefore ignore permutations of letters. PW is the same as WP; SPW, PSW, PWS, etc., are also all equivalent to each other.]

Of course, that's not the only way we could have decomposed the relationship. An equivalent verbal description of the ternary relationship is that when we order the part from one supplier we may have it sent to certain warehouses, while we may have other suppliers send the same part to other warehouses. Now we are thinking first of a binary relationship between parts and suppliers, which we can call "sorgs" (because I can't think of a good descriptive noun). Then we construct the binary relationship between sorgs and warehouses. This time we have decomposed the ternary relationship into the two binary relationships PS and W(PS).

And, in similar fashion, we can rationalize a third decomposition, into the pair of relationships WS and P(WS).

There are always multiple ways to decompose relationships of higher degree into binary relationships. We have seen the three possibilities for decomposing ternary relationships. For degree four, there are 15 different ways: PSWT can be decomposed into the forms P(SWT), S(PWT), W(PST), and T(PSW), with the ternary relation in each of these being decomposable three ways, plus the forms (PS)(WT), (PW)(ST), and (PT)(SW).

An example of the form (PW)(ST) is interesting to look at, because it illustrates how instances of relationships are themselves things which can be related to each other. We can take P and W to be parts and warehouses as before, with the relationship PW still called "allocations". S is still suppliers, and T is now truckers; PSWT is the relationship that certain suppliers use certain truckers to deliver certain parts to certain warehouses. There is a relationship ST between suppliers and truckers -- which suppliers use which truckers -- and we can arbitrarily call these "sources". A "source" is a combination of one supplier and one trucker. The relationship PSWT can now be expressed as the binary relationship (PW)(ST)

between allocations and sources: which source services which allocation. This new relationship is relating instances of two other relationships.

Simplicity

The reason we dwell so long on these decompositions into binary relationships is that some important data models are founded on such decompositions. From some points of view binary decomposition may be the best and simplest way to describe all relationships.

The main advantage of such decompositions is that they permit any relationship to be built out of a single kind of building block (the binary relationship), giving the description of all relationships a very regular format [Titman]. Hence, it is "simpler".

Whether or not you believe that depends on what you mean by simplicity. Simplicity is very much a subjective notion, and we can give it at least two interpretations. On the one hand, simplicity in a descriptive language (for data or programs or anything else) means minimizing the number of different terms that can be used in the description, thus minimizing the number of terms that have to be learned by users and implemented in the system. The regularity of binary relationships provides this kind of simplicity. According to this criterion, Roman numbers are simpler than Arabic, since they use fewer different characters [C&A 70]. For that matter, binary notation must be the simplest of all. Also, by this criterion, the simplest computers of all are Turing machines. Would you like to program your application on one?

On the other hand, simplicity is achieved when descriptions can be written concisely, and can be carried on paper and in the mind as single broad concepts rather than complex bundles of tiny concepts. This kind of simplicity is provided, for example, in the relational model, which can directly accept a definition of a PSNT relationship, rather than requiring a sequence of definitions such as NT, S(NT), P(S(NT)). This kind of simplicity is achieved at the cost of requiring the system to understand a larger variety of constructs (i.e., relationships of all possible degrees). It is often argued that this wealth of additional constructs is redundant and adds no new function, since everything can be expressed in terms of the "primitive" constructs, e.g., binary relationships.

We can further illustrate the concepts, and problems, in a system for specifying arithmetic expressions. Consider one language which has terms for expressing addition, division, and counting, and another language which has all these plus a term for "average". Both languages have the same functional capability for describing things. The first is simpler because it has fewer terms to be defined, learned, and implemented. But the second language is obviously easier to use if you want to compute an average. To average a LIST of values, one writes

```
X = SUM(LIST) DIVIDED-BY COUNT(LIST)
```

in the first language, and

```
X = AVERAGE(LIST)
```


in the second.

We might observe that each kind of simplicity is appropriate to a different level in the system. The first language is appropriate to a low, internal, implementation level of the system, where implementation cost is of primary concern. The second language is appropriate to a higher external interface which is seen by users. Some kind of transformation process is required between the levels, such as a language translator which takes the second statement above as written by a user and transforms it into the first statement above to be executed by the system. This corresponds, for example, to the implementation of a relational data base on top of an interface dealing only in binary relationships.

This split level approach has some disadvantages. Since the direct intent of the user is not transmitted to the underlying system, the system may not be able to optimize and perform the function in the best possible way. In the averaging example above, the system executing the first statement would take two passes through the list, once to accumulate the sum and again to count the elements. If the system understood "average" directly, it would do both in one pass.

In a nutshell, simplicity can mean either a small vocabulary or concise descriptions. Both have their value.

Incidentally, let me mention still a third kind of simplicity, which may be even more important than the other two in the area of data description. This is "familiarity". The easiest description system for you to learn and use correctly may well be the one which is closest to something you already know, regardless of how objectively complex that may be. It is precisely this phenomenon, for example, which makes the metric system of measurement much less simple for us (and most of my readers) to use, although it is obviously simpler by any objective criterion. The trouble with this approach, of course, is that it is subjective and degrades very much on who the users are. How do you measure it? And does it lead you to support a number of systems, each appealing to a different group of users? (There is this hazard: the apparent familiarity can also lead users astray. They are prone to making false assumptions about the system, in those cases where the system does not behave the same as the thing they are familiar with.)

Unnecessary Choices

Another concern arises from the fact that in describing an n-ary relationship as a composition of binary relationships, one has to select one of the many possible compositions. That is, one has to decide whether to specify PWS as P(WS), W(PS), or S(PW). There is often nothing in the way that users think of this relationship to prefer one form over the other two. Hence this arbitrary choice should not have to be made in the conceptual model of the information system. Also, there is a danger that this form of specification may become entangled in implementation and performance concerns. The specification P(WS) suggests an implementation in which people just interested in the relationship between warehouses and suppliers, i.e., WS, will get better service than those interested only in parts and warehouses, i.e., PW. Any such implication

in the conceptual model should be avoided. An administrator should be free to change the implementation structures and performance optimizations without having, for example, to respecify P(XS) as S(PX).

EXPLICIT AND IMPLICIT RELATIONSHIPS

Explicitly Defined Relationships

Explicit declaration of relationships permits their properties (such as those described in section 0.8,9.6) to be specified directly to the system, to be enforced independently of the representation of the relationship.

Unless names of relationships are known as part of the data content of the system, it is difficult to answer such queries as:

- "What relationships exist between x and y?"
- "In what relationships is x involved?"

When the manipulative interface is expressed directly in terms of named relationships, then there is considerable latitude in the manner of representing the relationships, with the alternatives being hidden from the user. Representation options include:

- Symbolic linkages via matching field values.
- Internal linkages such as pointers.
- Inclusion of such linkages with other attributes of the entities involved (e.g., pointing to or naming each other).
- Inclusion of such linkages in a separate structure (e.g., intersection records) which represents the relationship.
- Computational procedures, such as composition of other relationships, or comparison of attribute values. (Such procedures could only be used for inquiry, not for modification of relationships.)

With named relationships, the syntax and semantics of queries can be made simple and uniform, independent of the method of representing relationships. (Also, the form of the query is likely to be closer to the natural language form.) For example, an inquiry regarding an "is employed in" relationship could conceivably be handled by a procedure which searches employment history records, looking for the department to which the employee was most recently transferred. An assertion that someone "is employed in" a certain department could be handled by making a new entry in the employment history to the effect that the employee was transferred into that department today (or else the assertion might also provide an "as of" date). Some users would appreciate not having to know that this was the implementation of the relationship.

As another example, consider the inquiry "find all employees located in Stockton". In a record oriented model without adequate capability for naming relationships, the user is obliged to discover that locations are specified in department records. This user has to formulate a query which selects the records of departments located in Stockton, and then finds the corresponding employees. In Sequel [[Astrahan], [Chamberlin]), for example, the query would take the form

```
SELECT NAME
FROM EMPS
WHERE DEPTNUM =
      SELECT DEPTNUM
      FROM DEPTS
      WHERE LOC = 'STOCKTON'
```

In contrast, if the user were provided with a defined relationship named "located in", then he need not know whether location information is contained in employee records, department records, or division records. This user is simply interested in who works where; he need not be responsible for knowing current corporate practices regarding the centralization of divisions or departments. (I.e., he need not know the functional dependencies.) Furthermore, if these practices were to change (migrating the location attribute from one record to another), so long as someone was kind enough to modify the internal definition of "located in", then our user need never be perturbed by the change. He can continue to inquire about who works where just as he did before.

This approach works better for inquiries than for updates. To change an employee's location, one does have to know whether employees can move about independently, or are constrained by the location of their departments or divisions. This knowledge can be gleaned from the organization of the relations -- i.e., the functional dependences -- in a relational data base. It can also be specified in consistency rules in a model which describes relationships directly.

Named relationships need not be implemented by internal structures such as pointer chains. They could be provided by means of "macro" facilities in the interface languages. For example, a macro facility could conceivably be added to the Sequel language, whereby a macro named "located-in" could be defined to expand into the Sequel text illustrated above. An end user might then formulate his query as "find employees located-in 'Stockton'", without knowing or caring about macro expansions, functional dependences, pointer chains -- or even network vs. relational models.

Of course, that name "located-in" relationship could also be presented to the user in the tabular form of the relational model. There could be an interface at which the (apparent) existence of such a relation is maintained for the user, independent of the manner in which it has to be materialized from the real underlying data (cf. [Boyce]).

Having named relationships as an integral part of the model is much the same idea as perceiving the model as a set of functions [Feliaus]. The external user understands the information system by knowing the names and descriptions of the functions, the required arguments, and the expectable return values. The functions correspond directly to the user's semantic understanding of the information. The implementation of the functions is hidden from the user.

The "links" of [Deichmann] and the "selection structures" of [Earnest] also relate to the concept of specifying named relationships.

Implicit Relationships

There is a disadvantage to systems which deal only in named relationships. They limit the user to following paths which have been previously declared by a data administrator, and make it difficult to follow paths implicit in other data stored in the system.

If two entities are related to a third in any way, then that in itself comprises a relationship among the first two. One employee might work in the same department as another. The secretary of a department probably serves as secretary for each employee in that department.

Attributes can provide such links in the same way as relationships. If an employee works at a certain location, this implies that his department has someone working at that location. If we have a mechanism for establishing that two attributes are "in the same domain", then we can infer a relationship between two entities having the same value of such attributes. E.g., we could infer that a supplier and a warehouse are in the same city.

Both the domain and the role of the attributes must be considered, to avoid misunderstanding the significance of the implied relationship. If an employee was hired on the date his manager graduated college, we mustn't infer that they were hired on the same date, or born on the same date.

Other kinds of erroneous inferences might also be carelessly drawn. If a part is available from a certain supplier, and a warehouse is serviced by that supplier, we can't infer that the part is stocked in that warehouse. (And even if it was, it might be a different supplier who supplied that part to that warehouse.) This is the "connection trap" [Codd], whereby an erroneous inference may be drawn from the "join" of two relationships on a common domain. This is a user error, not the fault of the data model, in ascribing the wrong meaning to the results. The user error arises out of mistakenly taking such relational operators as "project" and "join" to be inverse operations, expecting that performing the two in succession returns the original information. A projection can decompose one relation into two; joining these two does not necessarily re-create the original relation.

One of the strengths of the relational model is that all such "implicitly defined" relationships are readily available, simply by joining relations on a common domain. It does require, however, that users correctly interpret the meaning of the joined relations.

In the relational model, all such relationships are equally "in" the system. One could view as a disadvantage, however, the fact that they are all implicit. Relationships are not specified, named, or described explicitly in the relational model. At best, they might sometimes be incorporated into the "column names" in the relations, at the whim of the person defining the relation.

A column name (like a field name in a record) really performs at least two functions ([Hall], [ANSI]): it should specify the domain from which the values may be drawn (e.g., dates or departments), and also the significance of its association with the entity identified by the key (e.g., date of hire, or department employed in). There is no formalism in the relational model to distinguish these two functions of the column name, and no discipline which discourages calling a column just "date" (or "hired") instead of "date of hire". [CodJ] suggests prefixing a role name to the domain name to distinguish two columns drawn from the same domain; [ANSI] recommends that "it is good practice for an attribute name to contain both the role name and the domain name". This is presumably only for the benefit of human readers. There is no defined syntax by which the system may extract the domain names, e.g., to verify commonality of domains for a join operation. Nor could a query processor follow a chain of relationships, where the names of the relationships were contained in the role portions of the column names.

CONCLUSION

It useful to examine the nature of information outside the context of any particular data processing technology. It is profitable to take a naive and open-minded look at the way such concepts as "entity" and "relationship" actually function in our (individual) views of reality.

It could lead to new objective assessments of the capabilities of current technologies, and point to new directions (and limits) for future technologies.

REFERENCES

[Abrial] J.R. Abrial, "Data Semantics", in [Klimbie].

[ANSI] ANSI/X3/SPARC, Study Group on Data Base Management Systems, Interim Report, Feb. 1975. Also in FDT (Bulletin of ACM SIGMOD) 7 (2), 1975.

[Astrahan] M.M. Astrahan and D.D. Chamberlin, "Implementation of a Structured English Query Language", Comm. ACM 18 (10), Oct. 1975.

[Boyce] R.F. Boyce and D.D. Chamberlin, "Using a Structured English Query Language as a Data Definition Facility", IBM Research Report RJ1318, Dec. 1973.

[C&A 70] "The Empty Column", Computers and Automation, Jan. 1970.

[Chamberlin] D.D. Chamberlin and R.F. Boyce, "SEQUEL: A Structured English Query Language", in [SIGMOD 74]. Also IBM Research Report RJ1394.

[CODASYL 71] CODASYL Data Base Task Group Report, ACM, New York, April 1971.

[Codd] E.F. Codd, "A Relational Model of Data for Large Shared Data Banks", Comm. ACM 13 (6), June 1970, pp. 377-387.

[DBTG] Same as [CODASYL 71].

[Douque] B.C.M. Douque and G.M. Nijssen (eds.), Data Base Description, North Holland, 1975. (Proc. IFIP TC-2 Special Working Conf., Mopion, Belgium, Jan. 13-17, 1975.)

[Earnest] C. Earnest, "Selection and Higher Level Structures in Networks", in [Douque].

[Felinus] J.J. Felinus, S.E. Madnick, and H.B. Schutzman, "Virtual Information in Data Base Systems", Report CISR-3, MIT, Cambridge, Mass., July 1974.

[Goguen] J.A. Goguen, "On Fuzzy Robot Planning", in [Zadeh].

[Griffith 71] R.L. Griffith and V.G. Hargan, "Theory of Idea Structures", IBM Technical Report TR87.559, April 1973.

[Griffith 75] R.L. Griffith, "Information Structures", IBM Technical Report TR83.013.

[Hall] P.A.V. Hall, J. Owlett and S.J.P. Todd, "Relations and Entities", in [Nijssen].

[Hayakawa] S.I. Hayakawa, Language in Thought and Action, third edition, Harcourt Brace Jovanovich, 1972.

[IMS] IMS/VS General Information Manual, IBM Form No. GH20-1260.

[Kent] W. Kent, "Describing Information (Not Data, Reality?)", IBM Technical Report TR03.012, May 1976.

[Klimbie] J.W. Klimbie and K.L. Koffeman (eds.), Data Base Management, North Holland, 1974. (Proc. IFIP Working Conf. on Data Base Management, Cargese, Corsica, France, April 1-5, 1974.)

[Metaxides] A. Metaxides, discussion on p. 181 of [Douque].

[Nijssen] G.M. Nijssen, Modelling in Data Base Management Systems, North Holland, 1976. (Proc. IFIP TC-2 Working Conf., Freudenstadt, W. Germany, Jan. 5-9, 1976.)

[SIGMOD 74] ACM SIGMOD Workshop on Data Description, Access, and Control, May 1-3, 1974, Ann Arbor, Mich., R. Rustin (ed.).

[Titman] P.J. Titman, "An Experimental Data Base System Using Binary Relations", in [Klimbie].

[Tsichritzis 75a] D. Tsichritzis, "A Network Framework for Relation Implementation", in [Douque].

[Tsichritzis 75b] D. Tsichritzis, "Features of a Conceptual Schema", CSRG Technical Report No. 58, University of Toronto, July 1975.

[Zadeh] L.A. Zadeh, K. Fu, K. Tanaka, and M. Shimura (eds.), Fuzzy Sets And Their Applications to Cognitive and Decision Processes, Academic Press, 1975.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

CURRENT ISSUES IN CONCEPTUAL SCHEMA CONCEPTS

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

/18/ M. Senko:
DIAM as a Detailed Example of the ANSI/SPARC Architecture,
in /3/

/19/ R. Weber:
A Semantic Model of Integrity Constraints in a Relational
Data Base, in /3/

/20/ P.P. Chen:
The Entity-Relationship Model,
ACM TODS, 1, 1, 1976

/21/ D. Sundgren:
An Infological Approach to Data Bases,
Urval Nr. 7, Statistika Centralbyran, Stockholm, 1973

/22/ H.A. Schmid, J.R. Swenson:
On the Semantics of the Relational Data Model,
Proc. ACM SIGMOD Conf. 1975

/23/ G.M. Nijssen:
An Exercise in Conceptual Schema Design,
Working Paper for the IFIP Working Group 2.6

/24/ E.F. Codd:
Further Normalization of the Data Base Relational Model,
In R. Rustin (ed.), Data Base Systems, Prentice Hall,
New York, 1971

/25/ J.R. Abrial:
Data Semantics,
In: J.W. Klimbie, K.L. Koffeman (eds.),
Data Base Management, North Holland, Amsterdam, 1974

Architecture and Models in Data Base Management Systems, G.M. Nijssen, (ed.),
North Holland Publishing Company, 1977

CURRENT ISSUES IN CONCEPTUAL SCHEMA CONCEPTS

G.M. NIJSSEN

Control Data
Brussels

Summary

One of the results of the IFIP TC2 Conference "Modelling in Database Management Systems", held in January 1976, is the acceptance of a DBMS gross architecture with three schemata, where the three schemata are defined as follows. The conceptual schema describes which information is considered to be within a specific Universe of Discourse, the external schema describes how a subset of this information is presented to a program, and the internal schema describes how the information is physically represented on storage.

The main result of the September 1976 meeting of IFIP WG 2.6 (Databases) was a consensus on a set of requirements which should be met by the concepts of the conceptual schema.

In this paper we will present a synthesis of various papers on conceptual schema concepts, presented at the 1976 IFIP TC2 Conference, and the results of additional research efforts performed during 1976.

Table of Contents

Introduction

The main themes at the IFIP TC2 Conference "Modelling in Data Base Management Systems" held in Freudenstadt, January 1976 (Proceedings, reference 10) were the DBMS gross architecture, probably better referred to as DBMS conceptual framework, and the concepts for the conceptual schema.

There was fairly general consensus at that conference with respect to a DBMS conceptual framework with three schemata and two transformations. We will briefly describe the main points of this framework in section 2.

Although the conference made some interesting progress with respect to concepts of the conceptual schema, it was very clear that one was quite far from consensus on the best set of concepts for the conceptual schema.

In order to make more progress in this area, the IFIP WG 2.6 (Databases) spent the major part of its September 1976 meeting on defining a set of requirements to be met by the concepts of the conceptual schema. Once there is consensus on a set of requirements for the concepts of the conceptual schema, one can more usefully discuss the sets of concepts which satisfy best these requirements. This set of requirements will be briefly discussed in section 3.

In section 4 we will present one of the sets of concepts that satisfies best the requirements for the conceptual schema concepts.

In section 5 we will present a technique to present information structure type and population diagrams. Such diagrams are useful in enhancing the communication on major parts of a conceptual schema.

In section 6 we will use the information structure diagram to illustrate the metaconceptual schema of the concepts presented in section 4.

Furthermore, in section 6 we will present a formal textual description of the metaconceptual schema of the conceptual schema concepts described in section 4. One of the remaining questions after Freudenstadt 1976 was: what is a semantically irreducable sentence? This is formally described in section 6 as part of the metaconceptual schema, while section 4 contains a simple decision table to determine the semantical irreducability of a sentence.

Section 7 contains a summary and conclusions.

2. DBMS conceptual framework or gross architecture

Databases are here to support information systems. For our purposes in this paper, it is essential to recognize that any input and output of an information system is information. In figure 2.a this is illustrated by the arrows entering and leaving the information system: each arrow represents information entering or leaving the information system.

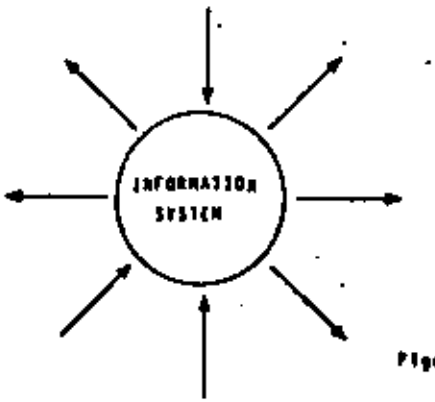


Figure 2.a

By a process called conceptualization (Schank, reference 13, 14) information is transformed into sentences, sentences are transformed into elementary sentences, and elementary sentences into object-role pairs (figure 2.b).

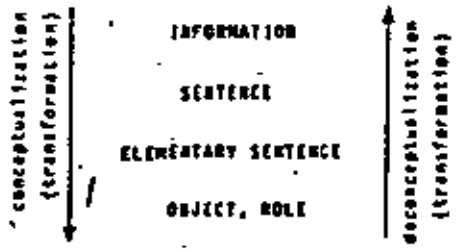


Figure 2.b

Information is a set of one or more related sentences.

A sentence is a set of one or more related elementary sentences.

An elementary sentence is a set of one or more object-role pairs, and each role appears at most once in the set.

Note. Information, sentence and elementary sentence in these definitions could all be qualified by the word "element" "instance", "occurrence" or "instantiation". (Burchhalt, reference 4)

What may enter and reside in an information system can be described completely, formally and simply by specifying the object types, the roles which an object type may play in an elementary sentence and restrictions that apply to a certain object type in a certain role. Such a description of object types, roles and restrictions is called a conceptual schema. A conceptual schema is like a prescriptive grammar: it describes which elementary sentences may enter and reside in the informationbase of the information system.

Every database (informationbase) contains a certain amount of information or knowledge.

A conceptual schema is a set of rules describing w h i c h information (or knowledge) may enter and reside in the informationbase.

An internal schema is a set of rules describing h o w the information (or knowledge) is phytically represented on storage media.

An external schema is a set of rules describing h o w information (or knowledge) is viewed by a program.

Having available these three schemata, one has in essence made a classification of all aspects of database description in three categories. One category, the conceptual schema, is only concerned with the conceptual aspects of the information of interest for a given Universe of Discourse; a conceptual schema may be regarded as an agreement between all persons involved in the database, and the machine. This agreement could be viewed as a set of rules which describe the sets of elementary sentences to be acceptable in the selected Universe of Discourse.
In the second category, the internal schema, one is exclusively interested in specifying of the elementary sentences described in the conceptual schema, the storage aspects, such as grouping, access paths, physical redundancy, etc.
In the third category, the external schema, one is dealing with a special view of the data, which is of interest for a specific application. There is a need for two transformations, one between the conceptual schema and the internal schema, the other one between the conceptual schema and the external schema. (Nijssen, reference 11).

A conceptual framework with the just mentioned three schemata and two transformations is in our opinion a prerequisite for more precise discussions regarding database matters.

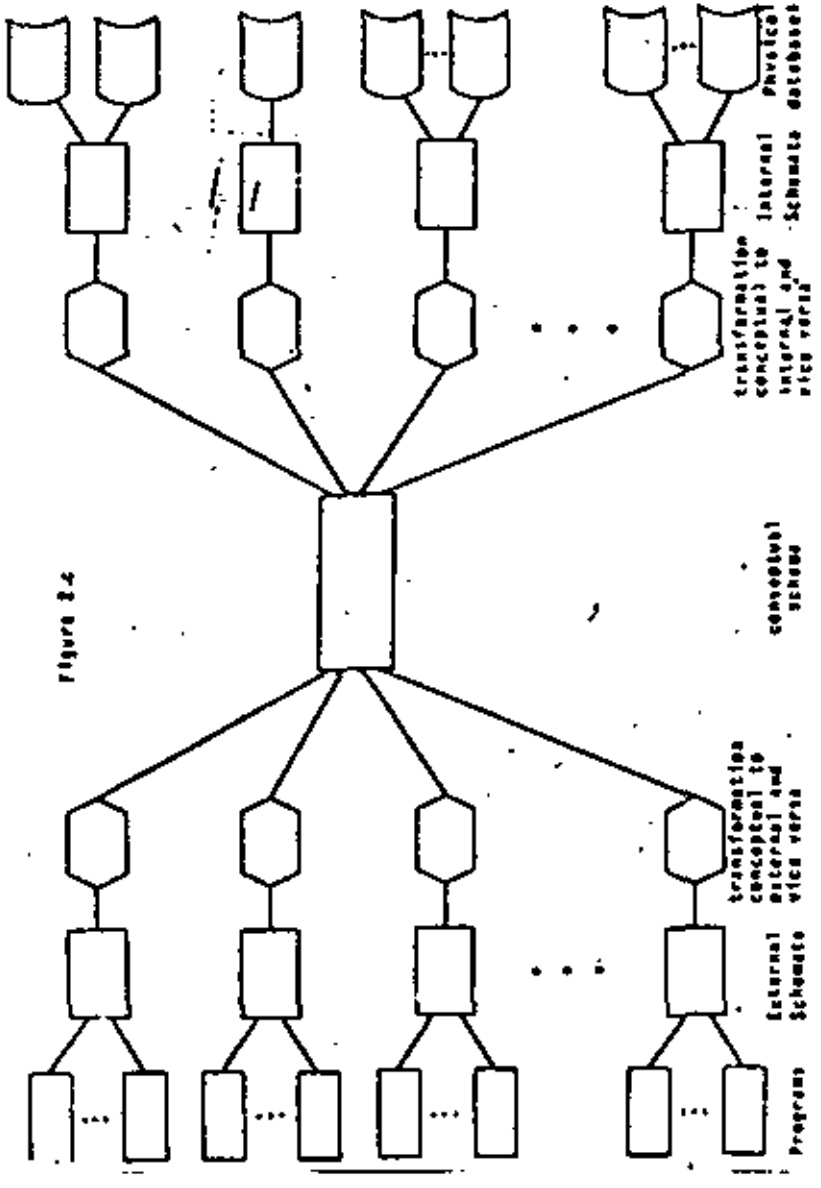


Figure 2.4

Figure 2.4 presents the major components of a DBMS gross architecture or DBMS conceptual framework. As can be seen in the figure, the conceptual schema is the central part of this framework. One could say that the external schemata correspond to surface structures associated with certain languages, while the conceptual schema corresponds to a deep structure. The conceptual schema as concept seems to be accepted by a rapidly increasing number of people; however the specific concepts to be used in the conceptual schema is still a matter of research and needs further discussion.

Before we go on to the next section, we would like to quote a few sentences from Artificial Intelligence literature: "A conceptual structure is defined as a network of concepts, where certain classes of concepts can be related to other classes of concepts. The rules by which classes of concepts combine are called the conceptual syntax rules. Since the conceptual level is considered to underlie language, it is also considered to be apart from language. Thus the conceptual syntax rules are organizing rules of thought as opposed to rules of a language" (Schank, reference 13, page 268).

"The elements must be defined on a level deeper than a semantic level that relates words to each other. We call this the conceptual level. The conceptual level is really not the same thing as the semantic level, since it has nothing whatsoever to do with a particular language" (Schank, reference 14, page 15).

3. Requirements for the concepts of the conceptual schema

At the end of the September 1976 meeting of IFIP WG 2.6, there was consensus that the set of concepts for the conceptual schema needs at least to satisfy the following requirements:

1. complete
2. formal
3. easy to
   - formulate
   - understand

4. easy to change

We like to add three more requirements, namely

5. easy to transform [to other views]
6. unique
7. orthogonal

Easy to transform is an additional requirement if we want to permit the COEXISTENCE of widely varying views or external schemata. The rules for describing a conceptual schema should be such that one situation cannot be described in more than one way, in other words it is unique. If there are two or more sets of concepts equally satisfying requirement 1 through 6, then the set with maximum orthogonality should be selected as the best one. [Nijssen, reference 12]

It is interesting to compare these requirements with requirements for the concepts to represent knowledge as discussed in the field of Artificial Intelligence. "We require of our meaning representation that it be unambiguous, unique and psychologically and computationally valid. Representations must be unambiguous since it is usually the case that the original meaning that the speaker chose to impart was unambiguous... By uniqueness, we mean that there can be only one way to express one meaning in the representation. By psychological validity, we mean that our model should conform as closely as possible to human processes..... By computational validity, we mean that computers should be able to operate with such notions in an efficient manner" [Schank, reference 14, page 15].

4. A best set of concepts for the conceptual schema

In this section we discuss "a" set of best concepts, which is not necessarily "the" best set, because it is not excluded that there may be more than one set of best concepts for the conceptual schema. We will present a best set of concepts which is an extension of the work of Falkenberg (reference 5, 6) and which is partly based on the Case Grammar of Fillmore (reference 3).

The model we present is still under development, on the other hand, it is stated with sufficient precision, internal consistence, and useful to guide further research.

Because this model is still under development, and it is based on the conceptual sentence structures underlying natural language, we have given it the name [RALIN, Evolving NAtural Language Information Model.

The [RALIN model contains the following primitive concepts:

- atomic object [elementary object]
- role
- name
- classification

and the following derived concepts:

- atomic sentence [elementary sentence]
- constraint

and the concepts type, population and instance are applicable to atomic object, sentence and name. In our words, a population is a set of instances which all belong to the same type. (See Borchholz, reference 4).

An atomic object is an object which is considered as an undividable entity for a selected Universe of Discourse.

A role is a function which an atomic object plays in an atomic sentence.

A name is a unique reference for an element in a given context.

Classification is an intellectual process which divides a set of elements in a set of disjoint subsets where all and only the elements in a subset share a common characteristic.

An atomic sentence is a set of one or more pairs, where each pair consists of an atomic object reference and a role reference, and each role appears at most once in the atomic sentence.

Constraints can be specified in various linguistic ways, such as procedural or declarative. It is our opinion that one should try to describe constraints in a declarative way such that a DBMS can "understand" the constraints. We are currently investigating the use and possible extensions of TAHALAH (Vandijck, reference 16) as a declarative constraints language for the conceptual schema.

When discussing the transformability of one view (data model) into another (or discussing the transformability among currently existing data models), it is very useful to give emphasis to two kinds of constraints, namely the "identifier" and the "functional subset".

An identifier of an atomic sentence type is a declaration of a set of object-role pairs whose value (concatenation of names) is unique within any atomic sentence population of that atomic sentence type.

In the traditional literature one often calls an identifier a "key" or a declaration to keep some elements "unique" within a population: no two sentences in a sentence population have the same value for a set of object-role pairs which constitutes an identifier.

A functional subset is a declaration which describes that the set of some object-role pairs must be a subset of another set of object-role pairs; the latter must constitute an identifier.

During the design of a conceptual schema one can easily determine whether a sentence is semantically irreducable (atomic, elementary) or not. A sentence is only semantically irreducable if it follows one of the four rules of figure 4.a.

number of roles in sentence	number of identifiers in sentence	number of roles in each identifier	rule
1	1	1	1
n (n>1)	1	n	2
n (n>1)	1	n-1	3
n (n>1)	>1	n-1	4

Decision table to determine semantical irreducability

Figure 4.a

In a Database Management System providing Coexistent different views, it is essential that while some users regard an atomic sentence, other users may regard the same atomic sentence as an atomic object; such sentences are called objectified sentences.

Note. In this paper, the words "atomic sentence", "elementary sentence" and "semantically irreducible sentence" are synonym references to one and the same concept.

Before we can give the formal description of a subset of the ERALIN concepts in itself, we want to introduce a diagram technique to enhance the communication on conceptual schema concepts.

5. Information structure type and population diagram

In database theory and applications, we need, besides a text oriented formalism, a convenient graphical formalism for representing such ideas as "atomic object type", "atomic sentence type", "identifier" and "functional subset" (all at the type level) and the same ideas at the population level.

For the ERALIM information model, we have tested since mid 1979 some alternative sets of graphical symbols in seminars on conceptual schema design; as a result of these experiences, we propose to use the following symbols for the ERALIM information model.

Atomic object type

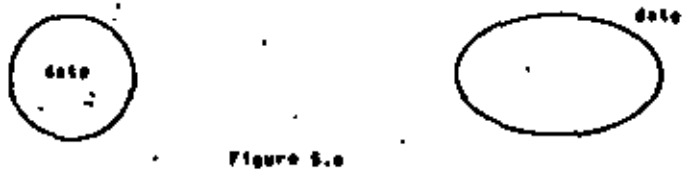


Figure 5.a

A circle, or ellipsis, containing a name, or having a name close to it, represents an atomic object type with that name.

Atomic sentence type

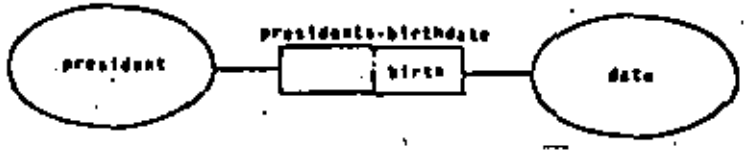


Figure 5.b

One or more rectangles, each rectangle connected via a straight line with exactly one atomic object type, and optionally containing the role which the atomic object type plays in this sentence, represent an atomic sentence type; each two rectangles have to share one common side. The name of the sentence type is placed above or under the rectangles.

Identifier

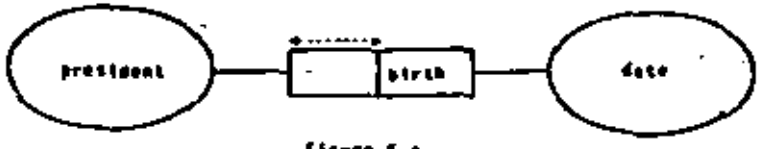


Figure 5.c

A dotted line, at each end having an arrow, ranging over some roles in an atomic sentence type, denotes an identifier.

With atomic sentences of two object role pairs, one has one of the following four possibilities with respect to identifiers.

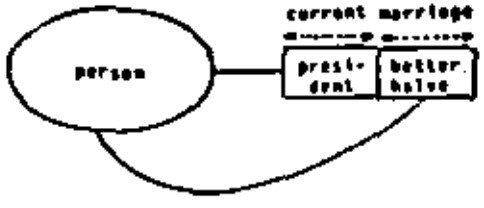


Figure 5.d.1

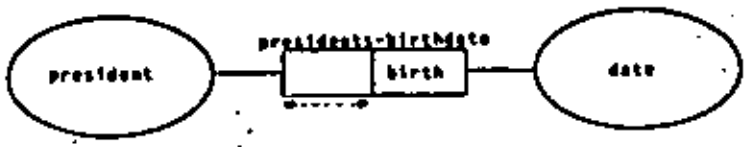


Figure 5.d.2

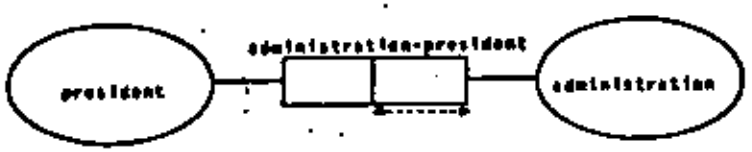


Figure 5.d.3

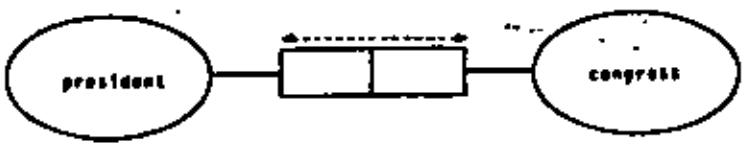


Figure 5.d.4

Note. Sid.1 satisfies rule 4 of the decision table in figure 4.a, 5.d.2 and 5.d.3 rule 3, and 5.d.4 rule 2.

Functional subset

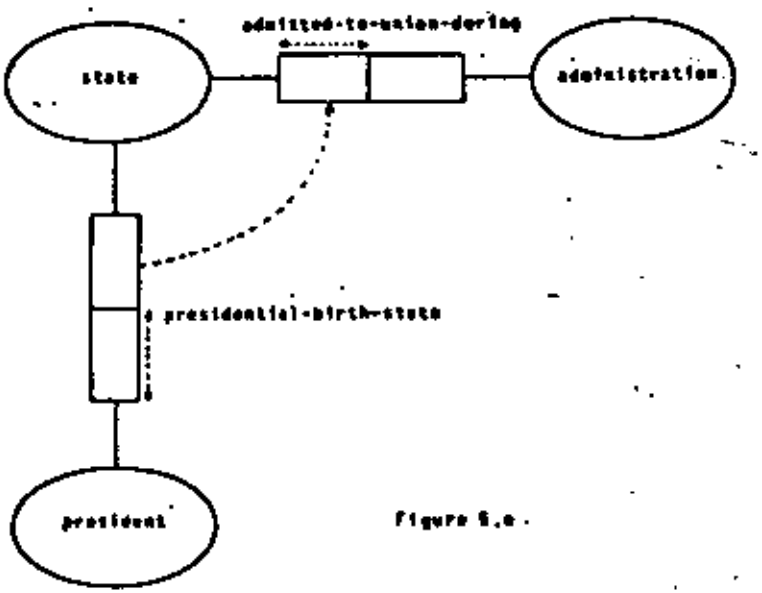


Figure 5.e.

A dotted line between two object-role pairs (or two compatenations of object-role pairs) with at one end of the dotted line an arrow, represents a subset, where the object-role pairs to which the arrow points is the range of the total function and the other object-role pairs, the domain of the total function.

For a functional subset, there is a syntax rule specifying that the arrow can only point to an identifier.

In figure 5.e the following is represented: the atomic sentence instance describing that a specific president is the native son of a certain state is only permitted to enter the Universe of Discourse (or database, or informationbase), if for this certain state the sentence instance is in the Universe of Discourse (or database, or informationbase) which describes that that certain state entered the union during a certain administration.

The symbols described so far refer to type declarations; one therefore could call these diagrams: information structure type diagram.

During the teaching of database concepts and implementation of databases, it become clear to us that type diagrams need to be complemented with population diagrams. The concurrent use of both type and population diagrams can help to increase the understanding.

For the information structure population diagrams, we propose to use the following symbols:

Population of atomic objects all belonging to the same atomic object type.

Figure 5.f

We propose, in analogy with the symbol commonly used in set theory, to represent a population of atomic objects as an ellipsis, containing a named dot to stand for each atomic object of the population; the name of the atomic object type is put outside the ellipsis.

Population of atomic sentences all belonging to the same atomic sentence type.

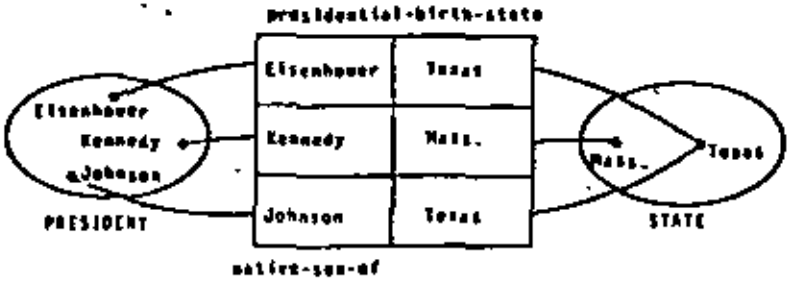


Figure 5.h

Two or more rectangles, each rectangle connected with exactly one atomic object via a straight line, and each rectangle containing the name of the atomic object playing a role in this rectangle, represents an atomic sentence instance; each two rectangles have to share one common side. If an atomic sentence population contains more than one atomic sentence instance, the instances are piled up. Optionally one can place the name of the atomic sentence at the top of the pile, and the roles at the bottom.

Identifier and subset

The symbols for identifier and subset used in the information structure type diagrams can be used in the same way in the information structure population diagrams.

In the sequel of this paper we will make use of these information structure diagrams if this can enhance the clarity.

Remark: A one argument sentence may be represented as in figure 5.i

Figure 5.i

Overlapping object types can be represented as illustrated in figure 5.j

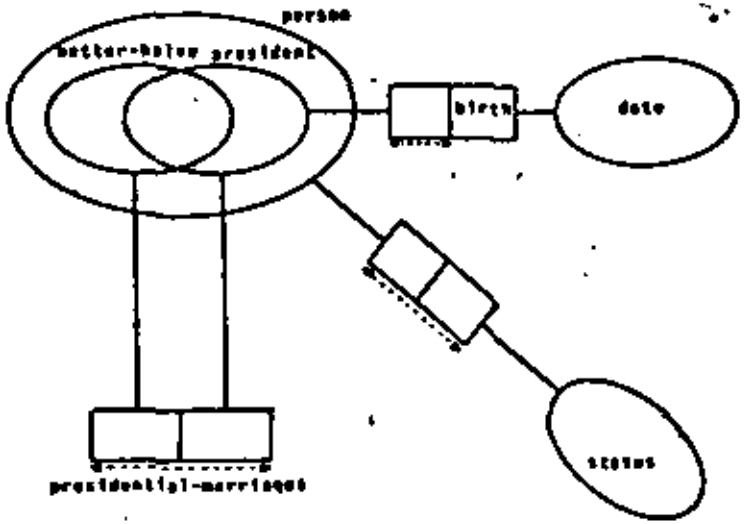


Figure 5.j

Objectified sentences can be represented as illustrated with an example in figure 5.k.

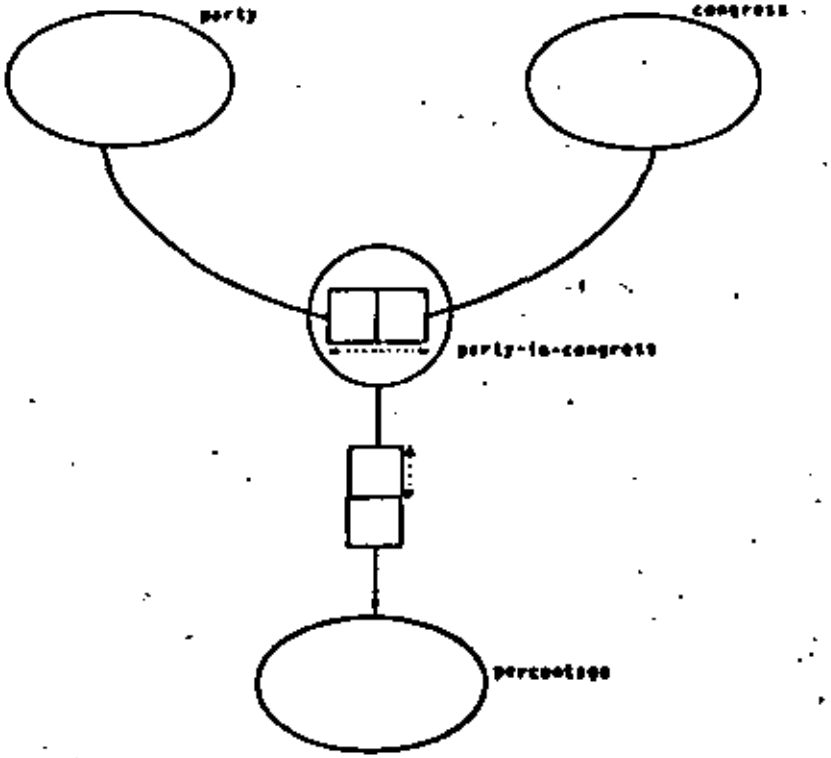


Figure 5.k

6. Formal description of a subset of a best set of conceptual schema concepts using the same concepts

In this section we will formally describe a subset of the best set of conceptual schema concepts as described in section 4 using the same best set and an information structure type diagram. As a consequence, the concept "semantically irreducable sentence" is formally defined. In figure 6.a we have represented an information structure type diagram of those concepts of the ENALIM model, which are of particular importance to describe the currently available data models.

Each role (nrole; n = meta) in the ENALIM model is associated with at most one object (nobject) and at most one sentence. (nsentence). Later in this section we will declare a constraint that each role must be associated with exactly one object and one sentence. (see constraint C1).

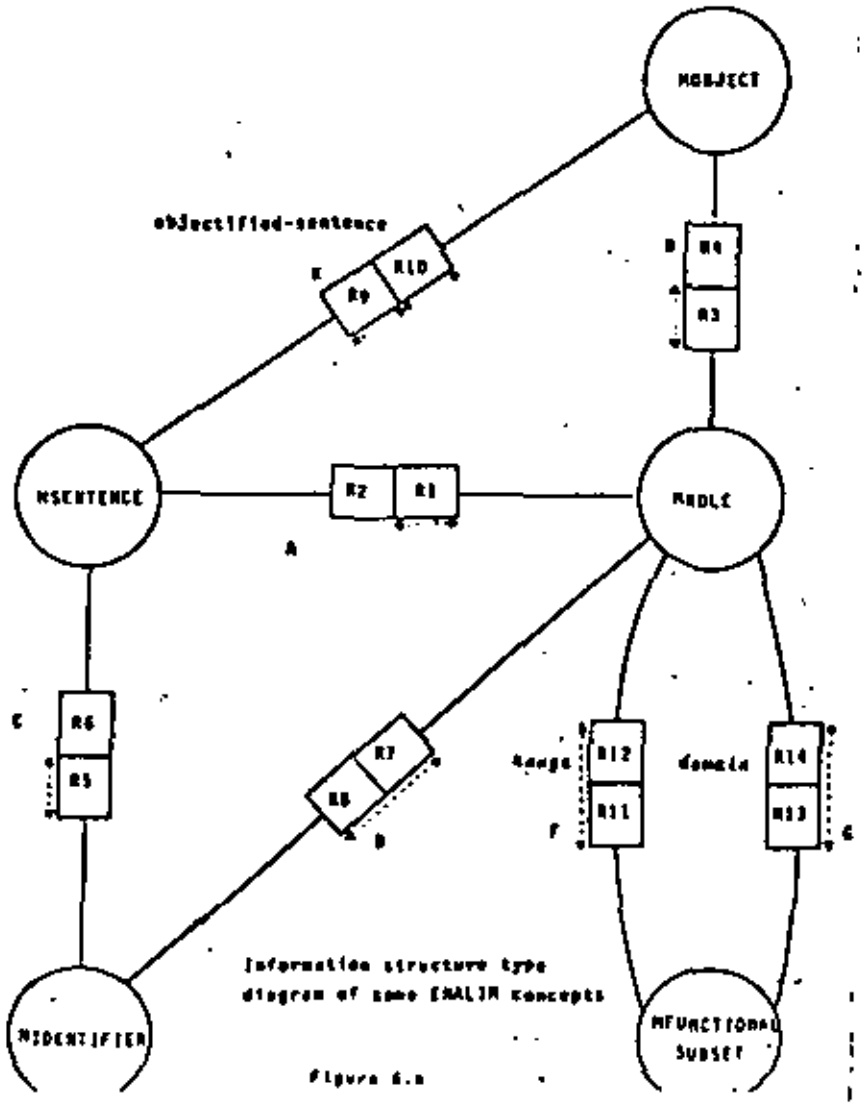
The semantical irreducibility of the atomic sentences in the ENALIM model is formally declared by the constraints under the comment "semantical irreducibility declarations". (see C5 and C6).

We would like to remark that most discussions on data models are restricted to three concepts only, namely sentence, role and object. This is somewhat unfortunate because it is also our opinion that constraints are an essential part of the model. (See also Schmid, reference [7]).

In the information structure diagram of figure 6.a we have included two kinds of special constraints, namely "identifier" and "functional subset". (Research is currently going on to bring in more declarative constraints). These two constraints play an essential role in transformability between various external user views.

With sentence E in figure 6.a is described that a certain sentence can be referenced as object by a role; such a sentence can function as object because it is an objectified sentence. There is a constraint saying that the identifier of such a sentence needs to cover all roles of the sentence. (see C13)

We will now present a conceptual schema to describe the ENALIM concepts using a language which has as concepts the ENALIM concepts.



Information structure type diagram of some ENALIM concepts

Figure 6.a

```
CONCEPTUAL SCHEMA NAME IS ERA[N-META-CONCEPTUAL-SCHEMA.

ATOMIC OBJECT DIVISION.

OBJECT NAME IS NOBJECT;
                NROLE;
                NSENTENCE;
                NIDENTIFIER;
                NFUNCTIONALSUBSET.

ATOMIC SENTENCE DIVISION.

SENTENCE NAME IS A
   REFERENCED OBJECT NAME IS NSENTENCE, ROLE NAME IS N2
   REFERENCED OBJECT NAME IS NROLE, ROLE NAME IS N1
       IDENTIFIER IS N1.

SENTENCE NAME IS B
   REFERENCED OBJECT NAME IS NOBJECT, ROLE NAME IS N4
   REFERENCED OBJECT NAME IS NROLE, ROLE NAME IS N3
       IDENTIFIER IS N3.

*/ THE PREVIOUS TWO SENTENCES ARE USUALLY THE MAJOR DISCUSSION
   DOMAIN OF THE CURRENT DEBATES ON DATA MODELS                 */

*/ WE WILL NOW DESCRIBE THE TWO SPECIAL CONSTRAINTS, NAMELY
   THE IDENTIFIER AND FUNCTIONAL SUBSET                        */

SENTENCE NAME IS C
   REFERENCED OBJECT NAME IS NSENTENCE, ROLE IS N6 .
   REFERENCED OBJECT NAME IS NIDENTIFIER, ROLE IS N5
       IDENTIFIER IS N5.

SENTENCE NAME IS D
   REFERENCED OBJECT NAME IS NROLE, ROLE IS N7
   REFERENCED OBJECT NAME IS NIDENTIFIER, ROLE IS N8
       IDENTIFIER IS N7, N8.
```



```
*/ THE PREVIOUS TWO SENTENCES DESCRIBE THAT AN IDENTIFIER
   CONSISTS OF ONE OR MORE ROLES, THAT A ROLE MAY BE A
   CONSTITUENT IN MORE THAN ONE IDENTIFIER, AND THAT AN
   IDENTIFIER BELONGS TO AT MOST ONE SENTENCE                  */

SENTENCE NAME IS G
   REFERENCED OBJECT NAME IS NROLE, ROLE IS N14
             */ DOMAIN OF THE FUNCTIONAL SUBSET                */
   REFERENCED OBJECT NAME IS NFUNCTIONALSUBSET, ROLE IS N13
       IDENTIFIER IS N13, N14.

SENTENCE NAME IS F
   REFERENCED OBJECT NAME IS NROLE, ROLE IS N12
             */ RANGE OF THE FUNCTIONAL SUBSET                 */
   REFERENCED OBJECT NAME IS NFUNCTIONALSUBSET, ROLE IS N11
       IDENTIFIER IS N11, N12.

*/ THE PREVIOUS TWO SENTENCES EXPRESS THAT A FUNCTIONAL
   SUBSET IS DEFINED BETWEEN N ROLES IN THE DOMAIN AND
   N ROLES IN THE RANGE.  IN THE ADDITIONAL CONSTRAINTS
   DIVISION (LATER IN THIS CONCEPTUAL SCHEMA) N IS SET
   EQUAL TO M.  (see constraint C14)                           */

*/ THE NEXT AND LAST SENTENCE IS DESCRIBED IN ORDER TO
   PERMIT THE USE OF SENTENCES AS REFERENCED OBJECT IN
   A SENTENCE.                                                 */

SENTENCE NAME IS E  */ OBJECTIFIED SENTENCES */
   REFERENCED OBJECT NAME IS NSENTENCE, ROLE IS N9
   REFERENCED OBJECT NAME IS NOBJECT, ROLE IS N10
       IDENTIFIER IS N9.
       IDENTIFIER IS N10.
       */ EACH OBJECTIFIED SENTENCE CORRESPONDS TO
          EXACTLY ONE OBJECT                                   */
```

```
ADDITIONAL CONSTRAINTS DIVISION.

CONSTRAINT NAME IS C1
    */ A ROLE MUST ALWAYS REFER TO EXACTLY ONE SENTENCE AND
       ONE OBJECT                                          */
    BEGIN.
    LET R1 OF A
              BE EQUAL TO
        R1 OF B
    END.

    */ C2, C3 and C4 ARE CONSTRAINTS ON IDENTIFIERS .      */

CONSTRAINT NAME IS C2
    */ EVERY SENTENCE HAS AT LEAST ONE IDENTIFIER          */
    BEGIN.
    LET R6 OF C
              BE EQUAL TO
        R2 OF A
    END.

CONSTRAINT NAME IS C3
    */ AN IDENTIFIER MUST CONSIST OF AT LEAST ONE ROLE     */
    BEGIN.
    LET R5 OF C
              BE EQUAL TO
        R6 OF B
    END.

CONSTRAINT NAME IS C4
    */ A ROLE WHICH IS A CONSTITUENT OF AN IDENTIFIER
       MUST BE A ROLE THAT BELONGS TO THE SAME SENTENCE
       AS THE IDENTIFIER BELONGS TO                        */
    BEGIN.
    LET CC BE C PARTITIONED BY R6
          */ CC IS AN INDEXED SET, WHERE EACH SET CONSISTS OF
             PAIRS (SENTENCE, IDENTIFIER) SUCH THAT THE
             SENTENCE VALUES IN THE PAIRS ARE EQUAL WITHIN
             A SET                                         */
```



```
        LET BB BE B
                WHERE RB OF B = R5 OF CC
            */ BB IS AN INDEXED SET, PARTITIONED BY R6     */
    LET AA BE A PARTITIONED BY R2
          */ AA IS AN INDEXED SET, PARTITIONED BY R6       */
    LET R7 OF BB
              BE SUBSET OF
        R1 OF AA
              WHERE R2 OF AA = R6 OF CC
                    AND
                  R5 OF CC = RB OF BB
    END.


*/ SEMANTICAL IRREDUCABILITY DECLARATIONS                  */


*/ THE FOLLOWING TWO CONSTRAINT DECLARATIONS, C9 AND C8
   FORMALLY DESCRIBE THE SEMANTICAL IRREDUCABILITY OF
   SENTENCES                                               */

*/ THE DECISION TABLE PRESENTED HEREUNDER IS TO BE SEEN
   AS ANOTHER AID TO CHECK SEMANTICAL IRREDUCABILITY OF
   SENTENCES IN A CONCEPTUAL SCHEMA COMPILER.              */
```

	Rule 1	Rule 2	Rule 3	Rule 4	Rule else
number of roles in sentence is n	n=1	n>1	n>1	n>1	
number of identifiers in sentence	1	1	1	>1	
number of roles in each identifier	1	n	n-1	n-1	
semantically irreducable sentence	x	x	x	x	
semantically reducable or otherwise incorrect sentence					x

Decision table for determining semantical irreducability of sentences .

Figure 6.b .


```
CONSTRAINT NAME IS C5
    a/ THIS DECLARATION CORRESPONDS TO RULE 1, 2 AND 3
       OF THE DECISION TABLE                              a/
BEGIN.
LET CC BE C PARTITIONED BY R6
       a/ CC IS AN INDEXED SET, PARTITIONED BY R6          a/
LET CC1 BE CC
            WHERE CARDINALITY OF CC = 1
       a/ CC1 IS AN INDEXED SET WHERE EACH SET HAS EXACTLY
          ONE ELEMENT, NAMELY A COUPLE CONSISTING OF A
          SENTENCE NAME AND IDENTIFIER NAME, AND THE
          SENTENCE HAS ONLY ONE IDENTIFIER                 a/
LET DD1 BE D
            WHERE R8 OF D = R5 OF CC1
       a/ DD1 IS AN INDEXED SET, WHERE THE ROLES ARE
          PARTITIONED BY SENTENCE                          a/
LET AA1 BE A
            WHERE R2 OF A = R6 OF CC1
       a/ AA1 IS AN INDEXED SET, WHERE THE ROLES ARE
          PARTITIONED BY SENTENCE                          a/
LET CARDINALITY OF AA1
                    MINUS
    CARDINALITY OF DD1
                    BE 0 OR 1
       WHERE R2 OF AA1 = R6 OF CC1
                    AND
             R5 OF CC1 = R8 OF DD1.
END.

CONSTRAINT NAME IS C6
    a/ THIS DECLARATION CORRESPONDS TO RULE 4 OF THE
       DECISION TABLE                                     a/
BEGIN.
LET CC BE C PARTITIONED BY R6
LET CC# BE CC
          WHERE CARDINALITY OF CC IS GREATER THAN 1.
```

```
LET DDR BE D
            WHERE R8 OF D = R5 OF CCR
        */ DDR IS AN INDEXED SET, PARTITIONED BY SENTENCE,
           BUT ONLY FOR THOSE SENTENCES WHICH HAVE MORE THAN
           ONE IDENTIFIER.                                  */
LET DDRI BE DDR PARTITIONED BY R8
        */ DDRI IS AN INDEXED SET, PARTITIONED BY IDENTIFIER
           WITHIN THE PARTITIONING BY SENTENCE AS IN DDR     */
LET AAR BE A
            WHERE R2 OF A = R6 OF CCR
LET CARDINALITY OF AAR
                    = MINUS
    CARDINALITY OF DDRI
                        BE 1
    WHERE R2 OF AAR = R6 OF CCR
                  AND
          R5 OF CCR = R8 OF DDRI
END.

*/ THE CONSTRAINTS C7, C8, C9, C10 and C14 ARE ASSOCIATED WITH
   FUNCTIONAL SUBSET                                        */

CONSTRAINT NAME IS C7
    */ THE RANGE PART OF A FUNCTIONAL SUBSET MUST BE AN
       IDENTIFIER                                           */
    BEGIN.
    LET FF BE F PARTITIONED BY R11.
    LET DD BE D PARTITIONED BY R8.
    LET FE BE FF
                WHERE R12 OF FF
                                IS EQUAL TO
                            R7 OF DD
    RECOMBINE FE
    RECOMBINE FF.
    LET FE MINUS FF BE EMPTY.
    END
```



```
CONSTRAINT NAME IS C8
    */ THE DOMAIN PART OF A FUNCTIONAL SUBSET
       MUST BE ROLES THAT EXIST                             */
    BEGIN.
    LET R14 OF G
                BE SUBSET OF
        R1  OF A
    END OF C8.

CONSTRAINT NAME IS C9
    */ THE ROLES INVOLVED IN THE DOMAIN AND RANGE PART
       OF THE FUNCTIONAL SUBSET MUST REFER TO THE SAME
       OBJECTS                                              */
    BEGIN.
    LET FF BE F BY R11
    LET GG BE G BY R13
    LET R4 OF B
              WHERE R3 OF B = R14 OF GG
                    BE EQUAL TO
        R4 OF B
              WHERE R3 OF B = R12 OF FF
    END OF C9.

CONSTRAINT NAME IS C10
    */ ALL DOMAIN ELEMENTS OF A FUNCTIONAL SUBSET MUST
       BELONG TO ONE AND THE SAME IRREDUCABLE SENTENCE       */
    BEGIN.
    LET G3 BE G BY R13
    LET A3 BE A
              WHERE R1 OF A = R14 OF G3
       */ A3 IS AN INDEXED SET WHERE A IS PARTITIONED PER
          FUNCTIONAL SUBSET                                 */
    LET A4 BE R2 OF A3
       */ A4 IS AN INDEXED SET WHERE EACH SET CONTAINS
          THE SENTENCE NAMES ASSOCIATED WITH THE SAME
          FUNCTIONAL SUBSET                                 */
    LET CARDINALITY OF A4 BE 1.
    END OF C10.
```

```
    */ CONSTRAINT C11, C12 AND C13 ARE CONSTRAINTS ON
       OBJECTIFIED SENTENCES                                    */

CONSTRAINT NAME IS C11
   */ A SENTENCE CAN ONLY BE USED AS AN OBJECTIFIED
      SENTENCE IF IT EXISTS                                     */
   BEGIN.
   LET R9 OF X
            BE SUBSET OF
      R2 OF A
   END.

CONSTRAINT NAME IS C12
   */ AN OBJECT WHICH IS AN OBJECTIFIED SENTENCE MUST HAVE
      AT LEAST ONE ROLE REFERRING TO IT                         */
   BEGIN.
   LET X10 OF X
            BE SUBSET OF
      R4 OF B
   END.

CONSTRAINT NAME IS C13
   */ A SENTENCE MAY ONLY BE OBJECTIFIED IF IT HAS AN
      IDENTIFIER WHICH COMPRISES ALL THE ROLES OF THE
      SENTENCE.                                                 */
BEGIN.
LET CC BE C PARTITIONED BY R6.
LET CC1 BE C
          WHERE CARDINALITY OF CC = 1
LET D01 BE D
          WHERE R3 OF D = R5 OF CC1
LET AA1 BE A
          WHERE R5 OF A = R6 OF CC1
```



```
LET AA2 BE AA1
        WHERE CARDINALITY OF AA1
                          MINUS
              CARDINALITY OF D01
                          IS ZERO
          AND R2 OF AA1 = R6 OF CC1
                  AND
              R5 OF CC1 = R3 OF D01

RECOMBINE AA2 INTO A2.
LET R9 OF X
        BE SUBSET OF
   R2 OF A2
END.

CONSTRAINT NAME IS C14
   */ THIS CONSTRAINT DESCRIBES THAT THE NUMBER OF ROLES IN THE
      DOMAIN AND RANGE PART OF A FUNCTIONALSUBSET MUST BE
      EQUAL.                                                    */
   BEGIN.
   LET GG BE G BY X13
   LET FF BE F BY R11
   LET CARDINALITY OF GG = CARDINALITY OF FF
      WHERE R13 OF GG = R11 OF FF
   END.
```

As a summary, one may say that a subset of ERALIN concepts is defined precisely by the above conceptual schema. This subset of conceptual schema concepts is useful to describe currently existing data models, and will also be used as a basis for further research.

7. Summary and conclusions

Now that a certain three schemata and two transformations architecture is becoming more and more accepted as a conceptual framework and the requirements for the concepts in the conceptual schema are better defined, it is time to start a more precise discussion on sets of concepts for the conceptual schema.
In this paper we have given a formal description of a subset of concepts for the conceptual schema. A major side effect of this formal description is a formal definition of the concept "semantically irreducible sentence". We hope that this formal description may contribute to a more precise discussion on the concepts for the conceptual schema.

Acknowledgements

We have benefited from interesting discussions with the members of IFIP WG 2.6 (Databases) and the members of the CDBMS implementation team at Control Data Europe. We want to express our indebtedness to John Owlett for indicating a major inadequacy in the description of semantically irreducable sentences during a discussion in WG 2.6.

References

1. ABIBA R. -- HELDBEL C. LEGRAND M.

A Unified Approach for Modelling Data in Logical Data Base Design

In: IRI

2. BENCI E. BOBART F. BOGAERT H. CABARES A.

Concepts for the Design of a Conceptual Schema

In: IRI

3. BRACCHI G. PAOLINI P. PELAGATTI G.

Binary Logical Associations in Data Modelling

In: IRI

4. DERCHOWITZ N.

Types and Related Concepts

Paper to be presented at ACM IEEE77, Liège, Belgium. Proceedings North-Holland Publishing Company.

5. FALKENBERG E.

Strukturierung und Darstellung von Information an der Schnittstelle zwischen Datenbankbenutzer und Datenbank-Management-System

Ph.-D. Thesis, University of Stuttgart, 1975.

6. FALKENBERG E.

Concepts for Modelling Information

In: IO

7. FILLMORE C.J.

The Case for Case

In: Universals in Linguistic Theory, E.Bach and R. Harms (eds) Holt, Rinehart and Winston, New York, 1968.

8. HALL P. OWLETT J. TODD S.

Relations and Entities

In: IO

9. MOULIN P. RANDON J. TEBOUL M. SAVOTSKY S. SPACCAPIETRA S. YAMBEW M.

Conceptual Model as a Data Base Design Tool

In: IO

10. NIJSSEN G.M. (editor)

Modelling in Data Base Management Systems

Proceedings IFIP TC-2 Working Conference on "Modelling in Data Base Management Systems", held in Freudenstadt, Germany, January 5-9, 1976.

11. NIJSSEN G.M.

A gross Architecture for the next Generation Database Management Systems

In: IO

12. NIJSSEN G.M.

An Exercise in Conceptual Schema Design

Working Paper, IFIP WG 2.6, October 1976.

13. SCHANK R.C.

The Structure of Episodes in Memory

In: Representation and Understanding, Studies in Cognitive Science, edited by D.G. Bobrow and A. Collins, Academic Press, New York, 1975.

14. SCHANK R.C.

Conceptual Information Processing

North-Holland Publishing Company, Amsterdam 1975.

15. SENKO M.E.

DIAM as a Detailed Example of the ANSI:SPARC Architecture

In: IO

16. VARDIJCK E.

Towards a More Familiar Relational Retrieval Language

Accepted for publication in Information Systems.

17. SCHMID H.A.

An Analysis of some Constructs for Conceptual Models.

In these Proceedings.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

THE TEMPORAL DIMENSION IN INFORMATION MODELING

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

20

[illegible] Models in Data Base Management Systems (S.B. Yao, ed.),
[illegible] Conference, 1977

THE TEMPORAL DIMENSION

IN

INFORMATION MODELING

Jan A. Bubenko jr*

Mathematical Sciences Department
IBM Thomas J. Watson Research Center
Yorktown Heights, N.Y. 10598

ABSTRACT

The problem of dealing with time-varying associations or relationships in conceptual information modeling is examined. A conceptual framework where time is treated in an unrestricted fashion is introduced. The binary and n-ary relational modeling approches are then discussed from this point of view. Also the paper comments on some approaches which include time as a basic concept in their frameworks. It is concluded that, when designing a conceptual schema, a time-unrestricted design level should precede the specification of a *finite*, time-restricted conceptual schema.

1 INTRODUCTION

According to the ANSI/X3/SPARC study group (AX3-75A) the 'conceptual schema' describes a limited (restricted) conceptual model of (parts of) the real-world (RW). This model is maintained for all applications. While several models for the conceptual schema have been suggested, very few have paid attention to the problem *how* to systematically design such a schema. In this paper we will not directly and explicitly address the process of designing and defining a conceptual schema. Instead, we will discuss modeling at the conceptual level and pay particular attention to the problem of dealing with *time-varying associations*. This is a highly relevant topic in modeling all RW systems as no system can be considered as static. By *temporal dimension* we denote the

---

* On leave from the Royal Institute of Technology/University of Stockholm, Sweden.

time aspect of conceptual information models of RW systems and we will discuss how time is treated, or can be treated, in various conceptual modeling approaches.

Dealing with dynamic systems is a common problem in engineering and science. The behavior of systems is conceptually modeled by sets of equations where time plays a fundamental role as an independent variable. Numerical treatment of these equations requires, however, that we transform the model to a discrete one which only considers a finite set of points on the time axis. We thus study the *state of the system at various points in time* and we may, for computational reasons, keep a 'history' of a set of model states at different time points in the limited computational storage.

Modeling of administrative systems has similarities with mathematical modeling of physical systems. The differences are that in administrative systems we have to deal with large and strongly varying sets of entities and that several aspects of the behavior of these systems are difficult to approximate by traditional mathematical equations. However, in both cases we wish to maintain a conceptual model of some system of interest. While mathematicians have approached the modeling problem from the existing mathematical framework, 'information scientists' have approached conceptual modeling of organisations and administrative systems from a 'computational framework' which initially was 'inspired' and restricted by existing computing and storage machines and concepts (for example sequential tape processing machines). Even today our computational resources are not unlimited. This fact has clearly influenced the development of conceptual modeling concepts for data base applications. The ANSI/X3/SPARC report (AX3-75A), for instance, deals in the *conceptual realm* with concepts such as *conceptual fields, conceptual groups, conceptual records and pieces* (not necessarily materialized). This should indicate the study group's view of a conceptual model as a finite and discrete data machine. While there are a few exceptions, most other conceptual modeling approaches have adopted this view.

In this paper we will therefore study how the temporal dimension can be incorporated at the conceptual level and survey how some earlier approaches have dealt with this problem. At the end we will discuss some consequences of not considering the temporal dimension at the conceptual modeling level. In our discussions we will,for illustration, use an extremely simple example which can be seen as a part of an inventory management application case. In spite of its simplicity, the case is general enough to illuminate problems of dealing with time at the conceptual level. First,however, we will outline a framework which includes the time dimension and which also facilitates a comparison of various modeling approaches.

2 A BASIC CONCEPTUAL FRAMEWORK

In this section a brief overview of our basic framework and its main realms will be given. A more detailed description is in preparation (BUB-76C).

The framework considers the following realms

The *real-world* realm(RW),which includes the concept of *object system* (OS).

The *conceptual model* realm, which contains notions of the *abstract* model (AM) and the *information* model (IM).

The *database* realm,which includes notions of *data structure* (DM) and *storage structure* modeling (SM).

The framework also includes descriptions of these models and the design processes (mappings) from one realm (level) to the next 'lower' realm. A particular method of information or data modeling is more or less explicitly concerned with all these realms,descriptions and processes. Most methods, however, focus their attention to techniques for definition and representation of information models or data structure models. It is obvious, also,that the use of a particular technique strongly influences the way the mapping/design processes above are carried out, i.e. the way models at different levels are developed.

The suggested framework rests, to a considerable extent, on an integration of notions and concepts for information modeling suggested earlier by the author (see for example (BUB-73C), (BUB-76A)) and others (see reference list). The most important of them is, we believe, the consideration of the *time dimension* in abstract information modeling suggested by Langefors (LAN-66A). The conceptual separation of *abstract* models and *information* models - or information modeling languages - as suggested in (BIL-76A) - has also helped to clarify certain conceptual issues. The time dimension in particular, as will be shown, facilitates a non-procedural and less restricted treatment and description of models at the conceptual abstract and information modeling levels.

The author would like to stress that the suggested framework does not claim to constitute a complete 'tool-box' for information modeling. The main purpose of presenting a crude outline of it in this paper is that it facilitates a discussion of certain modeling issues.

The real-world realm

The *object system* of an information system and its data base is the part of the real-world which is relevant to a particular application problem. Thus, the process of defining the relevant OS is essentially a *system analysis and design* activity which includes

- study of the actual organisation's goals, objectives and policies
- study and/or design of operations, flows(material or information) and decision processes.
- study of the organisations interaction with its environment
- determination of information needs and requirements
- study of the information system's 'socio-technical' properties and consequences
- specification of goals,objectives and requirements concerning the information system including its data base (including a statement of design criteria) etc.

Several techniques and methods for carrying out and documenting these activities have been published (see for example (CGR-74A) for an excellent survey and (LAN-74C) for a general framework). It is, however, beyond the scope of this paper to more in detail consider this realm.

The conceptual realm

The conceptual realm is seen to include two subrealms which are outlined below.

The abstract model realm

The organisation's (or physical system's - in an engineering type application) data base will ultimately carry information, represented by data, about an *abstract model* of the organisation (resp. the physical system). The following are our basic notions of this realm:

- *Entity:* an undefined concept, but which has been sufficiently well informally and intuitively described by others (for example (CYL-62A)). Initially, our real-world and object system discourse consists of an unclassified, varying set of objects (concrete or abstract). These correspond to a set E of entities in our abstract model.

- *Property:* an undefined concept, but we consider properties as a subset of E - our discourse of entities.

- *Concept class:* a concept class C_i is a subset of E, such that all entities of C_i have one property in common. This is the *defining property* of that concept class. Concept classes may be non-disjoint.

- *Association:* an association is a well defined relationship between a finite number of entities, where each entity plays a well defined *role* (possibly multiple roles). This relationship can, conceptually, hold for time intervals or only for discrete points in time. Associations with the same meaning, but concerning different sets of entities, constitute an *association type*. It is defined by $A_i(r_1:C_{i1};r_2:C_{i2};\ \dots\ ;r_n:C_{in})$, where A_i denotes the association type, r_j the roles and C_{ij} denotes concept classes. In some simple cases the roles are obvious and can be left out.

- *Event:* an event is an observation or decision in the object system at time t of the *beginning*, *ending* or *occurrence* of some association defined in our model. It is understood that an event is always related to a time point or interval - its occurrence or observation time. Furthermore, we can distinguish between *external* events and *internal* events. Both are parts of our model but external events 'lie on the boundary' of or system and can only be observed, while information about internal events can be theoretically (sometimes approximately) derived and possibly also measured or observed in the object system. As we can see, the distinction between them depends on where we have drawn the abstract model's boundary. Events can be triggered by other external events outside the control of the information system or by external events in combination with information about the actual state of the model.

- *Assertion:* At any time t we can assert that a particular association holds, i.e. is *true*. Thus, every assertion is related to its *asserting time* t. The expression $A_i(t)(r_1:e_{i1};\ \dots\ ;r_n:e_{in})$, where $e_{ij} \in C_{ij}$ is either *true* or *false* (or possibly undefined, f ex. if some of its components e_{ij} do not exist at t). Basically, and practically, we are interested mainly in true assertions.

- *Conclusion:* A *conclusion* is an assertion which can be proven (deduced) true on the basis of information about existing events and defined deduction rules. Consequently, every conclusion is associated with at least one time point or interval - the time when the conclusion was drawn.

- *Time:* Relationships and associations in the real world do frequently change. So do the set of participating relevant entities. It is necessary that these properties are given explicit consideration in the abstract model. The concept of *time* is therefore fundamental in the realm of conceptual models. This has been earlier recognised by some researchers in the area of information modeling (see for example (LAN-66A), (SUN-73A), (BEN-76A) and (FAL-75A)) and also by a few authors in the field of artificial intelligence (for example (KAN-75A)). In our abstract modeling framework time plays two kinds of roles: *extrinsic* and *intrinsic* (cf also (YOU-SRA)). The extrinsic time is the *time when* a particular assertion is made or conclusion is drawn. Whether this time relationship is explicitly recognized or not *it always exists*. Intrinsic time plays a role as part of the definition of an association type, i.e. it constitutes parts of its 'meaning'. An association type may or may not contain intrinsic time components. Consider the following examples: we draw a conclusion at time t_1: THE QUANTITY ON HAND OF ARTICLE XYZ IS 41. Here t_1 is extrinsic. Next, consider the assertion at t_2 about the above conclusion, $t_2 > t_1$: AT t_1 THE QUANTITY ON HAND OF ARTICLE XYZ IS 41. Observe that the latter is true for all extrinsic $t_2 > t_1$ and that t_1 has now 'moved to an intrinsic role'. Consequently, t_2 will also move to an intrinsic role if we assert at $t_3 > t_2$ the assertion about the first conclusion etc. In light of this we see that information about events can be seen as assertions abot associations (describing the event), where the event occurrence or observation time has moved to an intrinsic role. In our abstract model of some application we will normally have both associations assertions about which do or do not depend on the extrinsic assertion time. Those which do not depend on time represent 'stable facts' or observations in our model where an intrinsic time relation implicitly or explicitly always is present.

An abstract model is designed by integrating known and anticipated information requirements from system owners and 'local' users (BUB-76A). The design process implies abstraction of the object system and classification of entities, events and associations into classes and types. Description of the AM also includes statement of axioms concerning dependencies ,consistency and completeness of the above mentioned components of the model. It is desirable that the description of the abstract model is non-procedural.

The 'state' of an abstract model (or abstract state (BIL-76A)) at time t is by many authors considered to consist of those entities which at t 'exist' in the AM and those associations which at t are *true*. The IMCs and the DMCs are then designed to maintain a correct current abstract state. This 'storage and current time-slice oriented' view leads, already at the abstract level, to insertion/deletion /updating problems and restricts the model's ability to evolve as new information requirements in the object system arise. Our general notion of an abstract model can be informally described as

> *An abstract model is a non-decreasing set of assertions about OS-events, a set of deduction rules and the set of conclusions that can be drawn about entity existence and associations at all system relevant times (historic, current or future).*

For instance, a conclusion drawn at time t_1 may not be possible to draw at time t_2, where $t_2 > t_1$, but the fact that it could be drawn at t_1 may still be of importance to the problem. We will attempt to elaborate this notion in connection with an example in the next section of this paper. It is desirable that the description of an abstract model is non-procedural.

The information model

In the abstract realm we have focused our attention to the *substance* of the model, i.e. which entities, associations and events to consider and their relationships and dependencies. In agreement with (HIL-76A),(CHN-75A) and (SEN-73A) we consider the information model realm as an emphasis on *representation* and *user-referencing*. In this stage we have to decide how the user should refer to the various components of the model. The only way humans can do this is by the use of *names*. We have here to decide how to refer to individual articles, warehouses, persons, colours etc. We also need user-informative names for concept classes, association types, roles etc. Furthermore, it is often the case that the same entity can be referenced by several different reference expressions composed of external names. We do, in this paper, not take any position for a particular formal language for this purpose but conclude that the reference issues have to be solved at this level. We stress again that, at this level, data-structuring and accessing problems are not considered.

The distinction between the abstract realm and the information realm may not be easy to make in a practical design situation because the only way users can reference components of a model is by the use of names which stand for (are 1:1-related to) 'known' concepts in our discourse. As names also must be considered as entities, the referencing problems will normally also influence our abstract model design. For instance, the decision to refer to a person by his name,birth-date and birth-time or by his social security number may imply two different abstract models and different set-ups of concepts. We do, however, maintain the position that the design and definition of an *abstract* model can essentially be done without particular consideration to referencing and naming problems, i.e. without the introduction of sets of external, user-informative names.

Ultimately, defining an information model we have to decide by which types of information objects or functions to represent event and conclusion types of our model. An information object is an entity which is fully based on and describable by external names. Information objects which represent events are denoted *statements*. A statement, once issued, never becomes false as it represents the occurrence of an event. A conclusion, drawn at time t, can be 'materialized' by an information object. It can be seen as a 'theoretical observation' (derivation) of our model and as such considered as an event representable by a statement. As for events, we may distinguish between external and internal statements. The problem of 'conclusion materialization' is a central issue discussing the temporal dimension. We will return to it in section 3.

According to our view, new statements informing about external events in the object system are, conceptually 'inserted' in the model at time points $t_1, \ldots, t_{i-1}, t_i, t_{i+1}$ — etc. Let us denote by

$S(t)$ the finite set of external statements in an information model at time t.

$A(t \mid t_i)$ the set of conclusions that can be drawn at t_i (on the basis of $S(t_i)$ and given inference rules) concerning relevant associations $A_j(t)$ of our model for all relevant times

If $A(t \mid t_i)$ is assumed not to change in the interval $t_i \leq t_k < t_{i+1}$, then we have designed a 'stepwise changing' model of our object system. This seems to be the normal case for most administrative type information system models. In this case, the time points t_i represent 'state changes' and $A(t \mid t_i)$ describes the current system state at time points t_k. We will observe, however, that several 'system variables' cannot 'exactly' be modeled in this way, for instance the moving average sales quantity for an inventory item changes from t_i to $(t_i + dt)$ even without having new statements introduced in this interval.

The datalogical realm

This realm is next to focus on implementation, operational and efficiency problems of an information model. It corresponds to the *string, encoding* and *physical device* levels as defined in (SEN-73A). The types of design decisions that have to be made suggests two subrealms : one where one essentially deals with data structuring and one where storage allocation and lay-out problems are in focus.

The data-structure model realm

The conceptual realm is essentially un-concerned with procedural, processing and efficiency issues of the information system or the data base. The information model's conceptual view on information can therefore be - somewhat misleadingly - said to presume a 'machine' with infinite memory and infinite processing speeds.

When designing a data model (often based on and restricted by some available DBMS (Data Base Management System) - a *data - machine* - and its capabilities) these 'practical' issues (finite memory and finite speeds) must be considered. This implies - first of all - that we from a practical point of view will not be able to keep track of all 'historical statements' (for example 'transactions') about events signalling entity associations in the DS. Normally decisions are made to either shorten or aggregate the 'history' or to discard the statements and to maintain only the 'current' (read: as far as the system knows - *the last known*) *state* (set of conclusions) of the AM." When a statement, signalling an event in the OS, arrives ,this gives rise to - possibly complicated - data insertion, deletion and updating operations in the data model in order to maintain the models 'state' consistent and 'up-to-date'.

Thus, *a data model is conceptually based on the notion of a 'machine' with finite storage and finite operating speeds.*

Major decisions, designing a data model, concern

- How to represent statements and conclusions by data structures?
- How to group (cluster) and link data for efficient access and processing?
- Which statements and conclusions to store explicitly in the finite memory and which to declare as derivable - by a machine with finite speeds- when they are referenced?
- How 'actual' or consistent to maintain the explicitly stored information about statements and conclusions, which conditions to apply for triggering, state updating etc?
- etc.

Of course, other issues, such as data protection, user convenience and ease-of-use, system availability and maintainability must also be considered. While the data structure part of a data model can be described in non-procedural terms the complete description of a data model normally includes procedures for its operation and maintenance.

The storage model

The storage model is concerned with implementation and storage allocation details of a particular data model and has to consider in further detail, efficiency and operational problems. A deeper discussion of storage models is beyond the scope of this paper.

3 DISCUSSIONS AROUND AN EXAMPLE

In this section an example will be provided which both partly illustrates our *framework* at the abstract modelling level and serves as a basis for examining the temporal aspect of information modeling.

Suppose we are concerned with inventory management of *articles* which each are available at different *warehouses*. We are interested in keeping track of available *quantities-on-hand* (qoh) both 'per article' and 'per article per warehouse'. Events in the object system, which affect this information, are vendor *shipments* and *deliveries* to customers. Vendors and customers as such are outside our sphere of interest. They lie outside our system's 'boundary', we assume. Let us introduce the following concept classes (or entity sets):

- A article types, or articles for short
- W warehouses
- Q quantities

which correspond to property concepts 'being an article', 'being a warehouse' and 'being a quantity'. The following association types reflect our information requirements, we assume

AQOH(A,Q)
AWQOH(A,W,Q)

where AQOH denotes 'quantity on hand of article' and AWQOH denotes 'quantity on hand of article in warehouse'. Assertions concerning both associations are, according to our framework, associated with an extrinsic time reference. The roles played by members of the concept classes A,W and Q in these associations are obvious. They are therefore left out. Members of a concept class will be denoted by small letters e.g. $a \in A$, $w \in W$, $q \in Q$. An assertion AQOH(t)(a,q) is *true* if we can conclude or observe that there at time t exist q units in inventory of article $a \in A$.

In this model we also introduce two event types which concern observations of *shipment events* (SHIP) and *delivery events* (DEL) at time points t_i

e:SHIP(t_i)(A,W,Q)
e:DEL(t_i)(A,W,Q)

Graphically, our simple conceptual model can be represented as in figure 1. As we will present several model alternatives for this sample problem we denote this by alternative A.

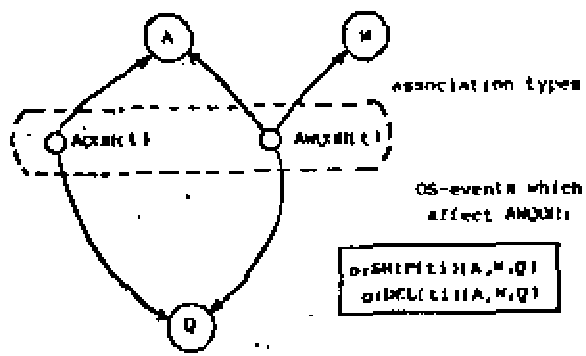


Figure 1
Model alternative A

We observe that AQOH and AWQOH change at time points t_i dictated by the occurrence of OS-events. In this model, information about the SHIP and DEL events directly affect the associations of type AWQOH and indirectly AQOH. The following equation and condition must hold for all times, warehouses and articles in inventory:

$$(\forall t)(\forall a)(\forall w)\{ AWQOH(t)(a,w,\_) = [\Sigma_{t_i \le t}\ e{:}SHIP(t_i)(a,w,\_) - \Sigma_{t_i \le t}\ e{:}DEL(t_i)(a,w,\_)] \ge 0 \}$$

Here we use the notation AWQOH(t)(a,w,_) to denote { q : AWQOH(t)(a,w,q) }. This condition puts a restriction on feasible DEL-events ; at no time can we deliver more than we have in inventory at a particular warehouse. It also illustrates the case when an event of type DEL is dependent on the current AWQOH-information status and the occurrence of some other external event, e.g. a 'customer order'. The customer order event and information about quantity on hand will, if the order-quantity can be satisfied, cause a DEL-event.

We also observe that the association AQOH can be derived from AWQOH:

$$(\forall t)(\forall a)[\ AQOH(t)(a,\_) = \Sigma_{w/a} AWQOH(t)(a,w,\_)\ ]$$

where w/a denotes those warehouses which have a particular article a in storage. Thus, according to the conceptual framework outlined in section 2 the quantity on hand can be considered as a single-valued discontinuous function of *time*, *article* and *warehouse*. Also, we may *conceptually refer* to this quantity at *any point in time* (and not only at state changes,for instance).

Using this conceptual framework it is also convenient to define other types of *derivable* associations, for instance

$$CHANGE\text{-}RATE(t)(a,w,\_) = (AWQOH(t-10)(a,w,\_) - AWQOH(t)(a,w,\_))/10$$

Clearly,other expressions are possible, for example

$$AWQOH(t)(a,\_,A)$$

which denotes a set of warehouses, possibly empty, which at time t had q units of article type a on-hand in inventory. We are now viewing warehouses as a multivalued function of t,q and a.

In those modeling methods for the conceptual level, which do not explicitly recognise the temporal dimension, information about the associations of types AWQOH and AQOH is normally treated as a finite set of *stored conclusions* which reflect the *last known* true assertions about these associations. The time dimension is not a fundamental concept of those approaches. Consistency rules require in this case that

$$(\forall a)[\ AQOH(a,\_) = \Sigma_{w/a} AWQOH(a,w,\_) \geq 0\ ]$$

Thus, the 'meaning' of the associations AQOH and AWQOH is now changed to *as far as the system knows ... the current quantity on hand*. Each time an OS-event of type SHIP or DEL occurs, these current quantities must be somehow 'updated' (for example by first adjusting AWQOH and then,for consistency, deriving or updating AQOH). This view is illustrated by our model alternative B in figure 2.

If we examine what we have done,by discarding the time dimension, we observe in this case that we have introduced *two new classes of entities* -we will call them *information objects* - which serve as means to store knowledge about the current status of two types of associations. 'Knowledge' is represented by associating each information object with adequate entities in the concept classes A, W and Q. These associations are *binary* and they can be given meaningful

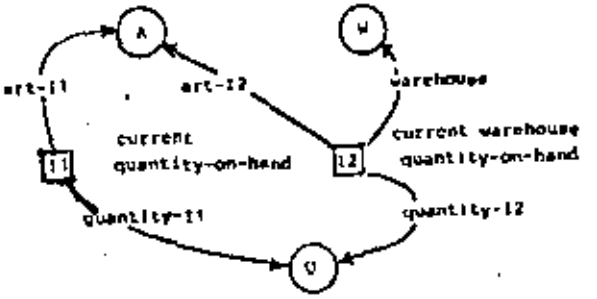


Figure 2
Model alternative B

names which describe their roles. Our 'transactions'- or statements about past SHIP and DEL events - are not part of this model alternative as we have made the design decision that the transactions in themselves are of no particular interest in our problem. Their only purpose is to 'update' the information objects. In figure 2 the information objects classes are represented by square boxes to mark our notion of them as entities. We note that, according to our assumptions, the following correspondences hold for I1,I2 and their associations in model alternative B

I1 → A is (1:1)
I2 → A x W is (1:1)
I1 → Q and I2 → Q are (M:1)

Choosing the model alternative B as adequate for our application problem we have tacitly also made the following assumptions

(a) the quantity on hand functions are constant between SHIP and DEL events

(b) there is no known requirement to be able to respond to queries where *time* is a vital parameter or queries which concern other system states as the current one,for instance:
   1 quantity-on-hand two weeks ago
   2 the average quantity on hand at (any) time t
   3 quantity on hand trends
   4 the difference in quantity on hand at times t_1 and t_2

Observe that we would not have any problems in formulating answers to these queries given our conceptual model alternative A. While most information or data modeling approaches would consider the model alternative B as 'conceptual' (or 'abstract' or 'logical') it is obvious that the decisions made and the 'conclusion materializations' introduced makes it unable to respond to queries given in (b) above.

It should, however, be obvious that we cannot -for practical reasons - realize (implement) information models with the full temporal generality of alternative A. For instance, we cannot in practice store requirements to be able to obtain responses to *on-line* queries concerning quantities on hand for any historical time point. So- at the interface between the information modeling level and the datalogical level decisions must be made which information about associations to support by representing it conceptually by a finite set of information objects. Note that we are at this interface not addressing datalogical design problems as we are not concerned with which information objects to *store* and which to *derive* or how to efficiently access and represent them in storage. For instance,information objects of type I1 in figure 2 are derivable from an associated set of objects of type I2 and the decision whether to always keep an updated version of I1 in storage or derive it when referenced can be postponed to the datalogical design phase or even to later phases. At this interface level we are rather concerned with which information objects to include in our model and how to deal with the time dimension. This is,in fact,the problem to design a *finite* model, processable by a finite machine, on the basis of an *infinite* model according to alternative A.

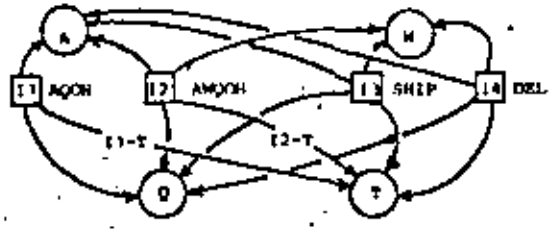


Figure 3
Model alternative C

In several applications, however, the information requirements are such that it is not sufficient to -conceptually- keep and maintain in storage only the current state of the associations. The ways to solve this problem depend on the model designer's insight and knowledge about potential user information requirements. One 'safe', but not always economical, way is to represent by stored information objects *all* statements (transactions) about external OS-events which are transmitted to the model. Time is an obvious parameter in these statements. In this way a *complete history* is maintained and responses can be generated to all conceivable queries which are derivable from the initial statements. Figure 3 (model alternative C) illustrates such a solution,where I3 and I4 are information object classes representing statements about shipment- respectively delivery events. Time is introduced as an infinite set of time points in class T. For simplicity the associations in alternative C need not be named to illustrate our point.

The number of elements in I3 and I4 is clearly *finite* and they form a binary-relational structure. Languages for navigating and retrieval in a *binary* structure,for instance the non-procedural FORAL (SEN-75C), are applicable to I3 and I4. We may also consider the I3 and I4 as 4-tuples with a well defined meaning constituting a subset of A x W x Q x T. This means that operations of *relational algebra or calculus* (COD-71B) can be applied to I3 and I4. Now, the situation

concerning I1 and I2 is slightly more complex. Whether we may consider I1 and I2 as finite sets of information objects depends on our definition of them. If we define I1 and I2 as *quantities-on-hand at time points when a change in quantity on hand for that particular inventory point occurred*, then the number of elements in I1 and I2 is less or equal to the sum of I3 and I4. If we, however, wish to represent by I1 and I2 the quantities on hand at *any* time t ∈ T then, clearly, their number of elements is infinite as T is infinite. Algebraic operations or binary navigations on I1 and I2 can now only be performed on *time restricted* subsets of them. If we by I1 and I2 wish to represent the *current* quantity-on-hand, then time (t='now') is implicit,the sets I1 and I2 are finite and we leave out the associations I1-T and I2-T in figure 3. Whether we decide to "keep" subsets of I3 and I4 as part of our model depends on anticipated information requirements.

We have above, in connection with model alternatives B and C, only discussed *some* possible ways to represent the (infinite) alternative A by finite model alternatives. Another alternative would be, if the requirements are such, to define function procedures for the 'quantities-on-hand' which had the arguments (A,T) and (A,W,T) respectively.

4 DEALING WITH THE TEMPORAL DIMENSION IN THE N-ARY AND BINARY RELATIONAL FRAMEWORKS

It might be interesting to discuss how the modeling problem according to the previous section could be dealt with within the conceptual frameworks of the n-ary relational respectively the binary relational approaches. We will start with the binary approach.

The binary approach

The binary approach is based on two fundamental concepts : the *entity* and the *binary association*. Modeling approaches based on these concepts have been suggested by Abrial (ABL-74A), Bracchi (BRA-76A), Bubenko et al (BUB-74A),(BUB-76A), Lindgreen (LID-74A) and Senko(SEN-75C). A possible 'schema' for the binary relational (or binary logical associative) approach is shown in figure 4. We observe that AQOH and AWQOH can be considered as 'entities' or 'information objects' only for a finite set of time points. Thus, navigating and selecting entities in the binary model can be carried out if AWQOH and AQOH are somehow restricted for a finite set of time points.

The concept of *derivation of entities on the basis of other entities* has not been explicitly paid attention to in the literature on binary models. In principle, however, this is equivalent to derivation of new relations (i.e. sets of n-tuples) on the basis of other relations in the n-ary relational approach.

Therefore we could define AWQOH and AQOH as sets of entities derivable from (and existence dependent on) the finite set of SHIP and DEL entities. These entities will carry information

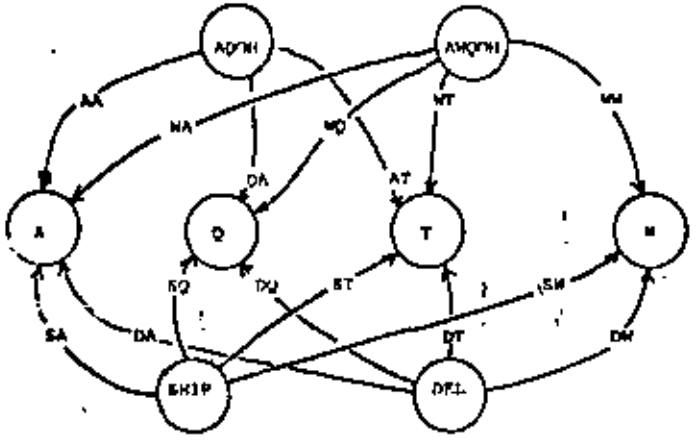


Figure 4
A binary relational schema. AA,WA,QA, ... etc are binary relation names

about quantities on hand at time points t ∈ T. To our knowledge there is no complete and generally accepted notation developed for expressing derivability,consistency and integrity relationships in binary models. Possibly can some non-procedural, query-oriented languages be extended and generalized in this direction (for example FORAL (SEN-75C)).

Using a calculus-like notation we could formulate the derivation of AWQOH-entities (for all time points) as shown below. We will denote by small letters a member of an entity class (for example awqoh ∈ AWQOH) and use association names as *selectors* (for example WA(awqoh) will select one element a ∈ A and DQ(del) will select a quantity (delivered) , q ∈ Q , if WA resp. DQ are functional).

```
(∀t)(∀awqoh)(WT(awqoh)=t ∧ WQ(awqoh) =
      SUM{SQ(ship:ship∈SHIP(awqoh,t))} -
      SUM{DQ(del:del∈DEL(awqoh,t))})
```

Here SHIP(awqoh,t) and DEL(awqoh,t) denote time- and association restricted subsets of SHIP and DEL entities as follows

```
SHIP(awqoh,t)={ship:SA(ship)=WA(awqoh) ∧
      SW(ship)=WW(awqoh) ∧ ST(ship)≤ t }

DEL(awqoh,t)={del:DA(del)=WA(awqoh) ∧
      DW(del)=WW(awqoh) ∧ DT(del)≤ t }
```


The function SUM has here an obvious interpretation. Clearly, SHIP(awqoh,t) and DEL(awqoh,t) denote subsets of SHIP resp DEL, which are associated with the same warehouse and article as the 'awqoh'-entity and which occured on or before time t. In the above example we have not paid attention to existential dependencies and other consistency/integrity constraints.

It is also possible to formulate the same derivation rule using a FORAL-inspired notation (SEN-75C).

```
DEFINE Q OF AWQOH
        (WHERE A = P1 AND W = P2 AND T = P3 )
= SUM Q OF SHIP
          (WHERE A = P1 AND W = P2 AND T ≤ P3 )
- SUM Q OF DEL
          (WHERE A = P1 AND W = P2 AND T ≤ P3)
```

In this definition P1 ∈ A , P2 ∈ W and P3 ∈ T are parameters. Using this derivation rule, a set of AWQOH-entities can now be created which are consistent with an existing set of SHIP and DEL entities.

The n-ary relational approach

A formalism for this approach was presented in 1970 by Codd (COD-70A) and it has subsequently been used by several others (see for example (ADI-76A),(BEN-76A),(CHN-75A),(HAL-76A)). Our previous discussion immediately suggests the following data model (relational schema)

```
SHIP(A,W,QSHIP,TS)
DEL(A,W,QDEL,TD)
AQOH(A,QOH,T)
AWQOH(A,W,QOH,T)
```

In this simple case, when all binary associations are functional, transformation to a n-ary model needs no normalization and is straight forward. SHIP and DEL are clearly relations in the 'traditional' relational data base approach sense. SHIP and DEL define a finite set of 4-tuples, where TS and TD are domains of shipment resp. delivery time points. Relational algebra or calculus can be applied to these relations. The relations AQOH and AWQOH are not relations in the finite, traditional sense if we define their interpretation as 'quantity on hand at time T' where T is a variable. In this case they may be considered as an infinite number of tuples (considering all possible values on the time axis). The problem how to deal with this 'infinite' situation has not been addressed in the n-ary relational data base literature.

The concept of *dividing* one relation from a set of other relations is, however, well known and either relational algebra or calculus are candidate tools for this purpose (sometimes so called 'library' functions may be needed to augment their expressive power). It is also well known that a data model- or data submodel - may contain definitions of relations, which are derivations from (possibly derivations from...etc) other relations (see for example (DAT-75A), p.115). This is equivalent to the concept of *user views* in n-ary relational systems under implementation (for example SYSTEM R (AST-76A)). Thus,we might consider AOOH and AWOOH as views with the understanding that they are materialized and treatable as finite sets of n-tuples only for a finite set of points in time.

Suggesting the concept of a 'time-view', we could define AWOOH in our data model definition as (using a mixture of SEQUEL (AST-76A) and DSL-ALPHA (see (DAT-75A)) -like syntax):

```
DEFINE TIME-VIEW AWOOH (A,W,Q,T)
RANGE SHIP S
RANGE DEL D
(FT)(ES)(ED) ( AWOOH.A = S.A ∧ AWOOH.W = S.W
        ∧ AWOOH.A = D.A ∧ AWOOH.W = D.W
        ∧ S.TS ≤ AWOOH.T ∧ D.TD ≤ AWOOH.T
        ∧ AWOOH.QOH = TOTAL(S.QSHIP) - TOTAL(D.QDEL))
```

TOTAL(.) is here an assumed library function. In order to have a view which can be operated upon by relational algebra or calculus the user now could define *time-restricted views* on top of this view, for example defining the current (T=NOW) quantity on hand as a *restriction* of AWOOH :

```
DEFINE VIEW CURRENT-AWOOH (A,W,Q)
SELECT AWOOH [ T = NOW ]
```

In the above definitions, the user is not concerned with how an implemented system maintains the defined views and restrictions. To our knowledge,however, there is to day no relational systems in existence which supports the above 'viewing' of information.

Considering an experimental n-ary DBMS, SYSTEM R, it is eventually possible that a view concerning the current quantities on hand at warehouses (CURRENT-AWOOH) could be defined by the use of SEQUEL-like operations (such as SELECT,COMPUTE,TOTAL) on tuples in the relations SHIP and DEL, as follows


```
DEFINE VIEW CURRENT-AWOOH AS:
SELECT A,W,Q
FROM S IN SHIP AND D IN DEL
COMPUTE: Q = Q1 - Q2
     COMPUTE Q1 =
     SELECT TOTAL(QSHIP)
     FROM SHIP
     WHERE A=S.A AND W=S.W;
     COMPUTE Q2 =
     SELECT TOTAL(QDEL)
     FROM DEL
     WHERE A=D.A AND W=D.W;
```

It is understood that this view is somehow automatically maintained - by the DBMS - consistent with existing tuples in the SHIP and DEL relations. In the SYSTEM R, the user could also himself control the maintenance of a relation,which is derivable or dependent on other relations, by the use of the TRIGGER-mechanism. For instance, in order to maintain a consistent set of stored tuples in the relation CURRENT-AWOOH, the user could define two TRIGGERS as follows (we describe them in a SEQUEL like syntax):

```
DEFINE TRIGGER XSHIP
ON INSERTION OF SHIP (
     (UPDATE CURRENT-AWOOH
     SET QOH = QOH + SHIP.QSHIP
     WHERE A = SHIP.A
     AND W = SHIP.W )

DEFINE TRIGGER XDEL
ON INSERTION OF DEL :
     ( UPDATE CURRENT-AWOOH
     SET QOH = QOH - DEL.QDEL
     WHERE A = DEL.A
     AND W = DEL.W )
```

In the above solution a problem arises when an inserted SHIP or DEL tuple has an <a,w> - pair, which does not match any 'existing' key in the relation CURRENT-AWOOH. This must be solved by introducing additional consistency and dependency checking and maintenance procedures. Note also that the designer-user now has full control over the degree of consistency in the data base. If he, for instance, is satisfied with with a 'correct' set of tuples in CURRENT-AWOOH only at 9 PM every day, then a triggering condition could be specified which activated a derivation process shortly before that time. This, however, would change the meaning of the corresponding relations.

Conclusion

Summarizing our discussion on modeling time-varying associations in the n-ary and binary-relational framework we have tried to demonstrate that neither of these approaches were originally intended for an abstract modeling level outlined in section 2. Both approaches view the information model for a real-world case as a finite,varying collection of n-tuples respectively a finite,varying collection of entities and binary associations. Time can,of course, be incorporated in these models by introducing time-domains in relations respectively time-entities and associations in binary models. We have also demonstrated that the notation and operations defined for models of this kind probably can be extended for use as definitional tools at the abstract, time unrestricted modeling level.

3 TREATMENT OF THE TEMPORAL DIMENSION IN SOME EXISTING INFORMATION MODELING APPROACHES

We have claimed that the majority of modeling approaches pay no explick attention to the temporal dimension. The information model of a particular application is seen as a finite ,varying set of information objects normally reflecting the current (last observed) state of a model of some real-world system. The state is changed by inserting,deleting and modifying information objects. Sometimes,in these approaches, rules are discussed how to define and maintain a set of information objects consistent and complete. There are,however, a few exceptions in this time-restricted and storage oriented approach. These will now be briefly discussed.

Young and Kent

Young and Kent (YOU-58A) have included the concept of time in abstract formulation of data processing problems. Time is,however, not used for defining information relationships but essentially for statement of time (delay) requirements concerning data processing and document production operations and for specification of document dates as functions of the times when the document is produced.

Langefors and Sundgren

The first approach which paid explicit attention to the problem of time in information models was presented in about 1965 by Langefors (LAN-66A). This approach has later been followed up by Sundgren (SUN-73A). Additional discussions can be found in (LAN-74B),(LAN-74C) and (SUN-74A).

The basic building block in this, so called *infological* approach, is the *elementary message* (e-message), a 3-tuple <o,a,t> which informs about an *elementary situation* (e-situation) in the object system. In an e-message, 'o' is a reference to a non-empty set of objects (entities), 'a' is a reference to a relationship type or to a property/value pair,and 't' is a reference to a point in time or a time period when this relationship or property holds or is measured (observed). Messages which inform about objects of the same classes and state properties respectively relationships of the same 'kind' are said to carry information of the same kind. They are said to conform to the same *elementary concept* (e-concept). There exist relationships between e-messages of a particular infological model in the sense that some e-messages can be produced from other e-messages, given a set of production rules. These relationships are called *precedence relationships*. Basic subsystems of an infological data base are considered the *schema*, the *nucleus* and the *filter*. The schema defines the particular object system model by enumerating all e-message types (e-concepts) and it also contains a set of message derivation rules. The nucleus is the set of explicitly stored e-messages and the filter function serves to protect the data base from false, meaningless or inconsistent messages. The nucleus may be *redundant* in the sense that some of its messages may be defined as derivable from other nucleus messages.

While time constitutes a fundamental part of the e-message, the references cited above (LAN-... and SUN-...) lack a complete notation and clear examples showing the author's basic intentions how to treat and reference the temporal aspect of information modeling. Our interpretation of their intentions follows.

The proposition *any message in an information system will be obtained either by observation in the object system or through the execution of a process which then takes other e-messages as input* (LAN-74C), indicates that their infological model is basically seen as a finite collection of e-messages. Also Sundgren's discussion of 'target sets' (p. 95 in (SUN-73A)) and associated examples reflect an, essentially, finite view of infological models. The contents of an 'infological data base', M(t), is at time t a set of known e-messages (SUN-73A). Some messages are explicitly stored in the nucleus while others are 'virtual' and derived when desired. Insertion and deletion of e-messages may only concern the nucleus. Each message in the infological data base is conceptually seen as having several *time-versions*. A new time version of an e-message is created when an e-situation in the object system occurs which concerns the particular <o,a> - pair. As some messages are derivable, it follows that an e-situation may, conceptually, introduce new time versions for several messages. Thus, an infological data base contains a finite set of e-messages, where each e-message may have several time versions. Each version signals a change in the property or relationship of a set of objects <o>. Conceptually the set of e-messages,including their time versions, is 'ever growing'. According to (SUN-73A), the number of time versions per e-message which the user considers to be of interest may be an important design parameter for the datalogical design phase.

Discussion

Our conceptual framework is to some extent inspired by Langefor's approach of dealing with time. Therefore similarities can be found between both conceptual views. Some differences are discussed below.

The concept of time versions appears quite natural when applied to *attributive* type e-messages, i.e. messages '<o,a,t>, where 'a' is an attribute-type/attribute-value reference. We then can think of time versions $<o,a_1,t_1>$, $<o,a_2,t_2>$, ... etc. which constitute full or partial history of an object o_i with respect to attribute (property) 'a'. An example of this situation is the e-concept <<article>,quantity-on-hand>, where we may think of e-message versions $<a_i,q_1,t_1>$, $<a_i,q_2,t_2>$,

... etc. where $t_2 > t_1$. Clearly, this set of versions describes the 'e-variation' for a particular article 'a'.

The situation becomes less clear when we talk about *relational* type e-messages, i.e. messages where 'a' denotes a relationship type. In this case, what changes - from time to time - is the set of objects referenced by 'o' and it is conceptually possible to think of the 'time versions' as being associated with every proper subset of objects of 'o'. The concept of 'saving the n latest time versions of a particular e-message type '([SUN-73A], p.139) does not seem particularly clear in this case, unless one also specifies for which subset of objects in 'o' this 'saving' has to be done. An example of this situation is the e-concept <<article,project,supplier>,SUPPLIES>, where SUPPLIES denotes a relation with well known interpretation from the n-ary relational world. If we have an e-message

$$<<a1,p1,s1>,SUPPLIES,t_1>$$

and subsume for it a new message at $t_2 > t_1$

$$<<a1,p2,s1>,SUPPLIES,t_2>$$

then this new message can be seen as a new relationship from the view-point of participating object subsets <a1>,<s1> in <a1,s1>. At the same time the set of SUPPLY-relationships where s1 participates is changed. Another consequence of this time-version approach is that in query references like $<a,<A_p,?>,t>$, if t is not the time of a historical time version of this message, then an adequate time version with a time reference 'nearest' to t must be located. Thus, a distinction is introduced between the contents of a data base and retrieved information.

Falkenberg

Falkenberg's *information sphere* (FAL-75A) consists of a set of associations, which each exist for limited time periods. An association is a semantically relevant set of objects $<E_1,E_2, \ldots E_n>$, where each object plays a specific *role* in the framework of the association. An association has a *beginning* and an *end* which are considered as *events* on the time axis. Specific representations (without regard to physical representations) of associations are denoted *significations*. *Building up and maintaining a data system means the subsequent input of all significations of all events happening in the information sphere.* In (FAL-75A) there a set of language operators are defined by which answers to queries about time-dependent relationships or associations can be obtained. The signification concept is also applied to query formulation.

In a later paper (FAL-76A) the time dimension is not given special attention. This paper,however, discusses semantic rules of 'data systems' and mentions concepts such as association dependencies (of the consistency type) and deduction of new associations.

Falkenberg's conceptual view on the temporal dimension is on several issues similar to our conceptual framework exemplified earlier. The main difference seems to be that Falkenberg considers the conceptual information model of some application as a finite, stored and varying collection of non-redundant 'facts' about events (associations). Consequences or conclusions from this set of facts seem not constitute a part of the model but are seen as derived responses to queries formulated with a relatively large set of operators, some of which are 'time-orientated'.

Bencl et al.

In their paper (BEN-76A) Bencl et al. discuss *concepts for the design of conceptual schema*. Their assumption is that the real-world can be completely described with *objects* (entities), *properties* and *associations* between these objects. The n-ary relational formalism is considered adequate to describe a conceptual data base. The existence of an object or an association is represented in the conceptual structure as (the existence of) a n-tuple of a relation. A conceptual data base is seen as a set of indexed relations $R = \{R_1,R_2, \ldots, R_n\}$. A conceptual data base R^{t+1} is obtained by applying to R^t operations, which conform to defined *e.g. evolution rules*.

The representation of time (time points or periods) in the conceptual structure is simply done by introducing, when found necessary, data types, e.g. one or more time domains. In the relation representing a set of objects or a set of associations. Of course, the exact meaning of these time-domains in the actual context has to be defined. The authors conclude that the introduction of time in a DDL causes no specific problems.

Discussing consistency in an evolving data base, the authors observe that the consistency of a data base does not have to be absolute at all times. This means that, in a practical DB-implementation case, we may permit that certain associations are not reflecting the current correct values according to initial statements about OS-events. In order to save computing power, these associations are periodically or conditionally 'updated' and not 'immediately' following some event which affects them.

Thus, in the approach of Bencl et al., time is incorporated in the conceptual model when found necessary (or natural) as data items (domains) in relations and a data base, conceptually seen as a finite collection of n-tuples of assorted degrees and interpretations, is maintained 'partly' consistent (at time point t) by updating operations triggered periodically or conditionally. From an information *modelling* point of view, this approach to definition of a conceptual schema brings it closer to the datalogical design level than to the abstract, time-oriented view outlined above.

6 CONCLUDING REMARKS

Suggested information modeling methods in the literature can, with respect to handling of the temporal dimension be grouped in two categories

1. Methods which more or less ignore the time aspect. A particular information model is here seen as a finite collection of information objects (statements,'facts'), which represent the current state, i.e. the set of true assertions made the last time information about some object-system event was inserted. Most modeling approaches, including the binary and n-ary relational, follow this underlying conceptual view. Of course, time can be, in a restricted sense, included in models following these approaches.This is, however, done in an

ad-hoc (arbitrary) way' by the model-designer by introducing intrinsic time domains in relations or by introducing entity-time associations in binary models.

2 Methods which include the time dimension as a fundamental concept of their framework. Langefor's and Sundgren's approaches fall into this category where an information model at the conceptual level is seen as an 'ever-growing' set of e-messages (may be considered as information objects) with different time versions. Also Falkenberg's approach, considering association beginning and ending events, belongs to this category. A problem with these approaches is, however, that none of them have elaborated and exemplified a complete notation for dealing with the definition and referencing of time varying associations and their dependencies or relationships.

Another issue, which concerns time-unconstrained (infinite) information models is the design problem to *restrict* them to a finite and state oriented model. It is clearly not possible to implement a model with the *full* temporal generality and therefore decisions must be made which states to maintain or which 'time versions' of information objects to support. This discussion suggests that we should consider two *sublevels* of the abstract (conceptual) information modeling level: one where we consider the time dimension in its full generality - leading to an 'infinite' model, and one where restrictions are decided to make the model *finite* and transformable to an implementable datalogical model. A method or 'strategy' how to transform an infinite model to a finite one has, however, not been addressed in the literature.

Studying published works on information modeling clearly shows that the temporal dimension has 'bothered' several researchers. However, after stating the importance of the temporal dimension, this issue is somehow dropped and the 'finite sublevel' of conceptual modeling is directly addressed. Summarizing our discussions, we believe that this 'short-cut' may have the following consequences.

1 A 'storage-oriented' view is enforced : instead of defining functions and time varying relationships the view is focused to storage and maintenance of function and relationship 'values'.

2 Decisions *how* to introduce time in the model, *which* time versions of information objects to maintain etc are made more arbitrarily than if a time-unrestricted model was used as a basis.

3 If time restrictions are introduced *while* creating a conceptual model then this makes it less convenient to formulate and integrate end user information requirements (designing a shared view based on different external schemata). Time-unrestricted ("infinite") models are, from a *problem* - oriented point of view, 'cleaner' and not concerned with processing (storage,updating,deletion) and efficiency issues. The end user can view the object system in a *real time perspective* and express information requirements and relationships referring to any time point and not referring to a finite,stored collection of messages,n-tuples or entities.

What we have pointed out above in no way eliminates the need and importance of 'finite' information models. It is only finite models which can be mapped to databases, models and which we can manipulate with relational algebra or navigational operators. Also, processing and efficiency issues can only be discussed starting out from the finite modeling level.

While methods based on the binary or n-ary relational view are significant achievements (in the sense of *problem orientation*) in information modeling over the stored data structure oriented views, represented by the hierarchical or network data structure manipulating machines (for example (CYL-71A)), they were not aimed for the time-unrestricted abstract modeling level. These approaches and the 'theories' around them are based on a finite view and the time - dimension problem is not always solved simply by adding time domains or entities. A new conceptual view and framework is needed, which,however,superficially may use some of the notational formalism introduced by these approaches. It is believed that the framework outlined in section 2 offers considerable generality in dealing with time which should make it a candidate basis for high level information modeling method development. Finally, it must be stressed that we do in no way suggest that the time-unrestricted level is suitable level for the 'conceptual schema' as defined by (ANSI-75A). We consider this level more as a 'design *phase*' and a basis which should logically precede the design of a *finite*, conceptual schema, which for processing and efficiency reasons must be 'strongly time-restricted' - at least for some years to come.

ACKNOWLEDGEMENT

The author wishes to thank Dr. Michael E. Senko at IBM Research for valuable criticism and suggestions.

REFERENCES

ABL-74A Abrial J.R.: *Data Semantics*, in (KIM-74A), pp. 1-60.

ACM-76A ACM Sigplan/Sigmod. *Proceedings of the Conference on Data : ABSTRACTION, DEFINITION AND STRUCTURE*. Salt Lake City, Utah,March 22-24, 1976.

ADI-76A Adiba M., Leonard M. and Delobel C.: *A Unified Approach for Modeling Data in Logical Data Base Design*, in (IFIP-76A), p. 634ff..

AST-76A Astrahan M.M. et al.: *SYSTEM R : a relational approach to data base management*, IBM Research, Report RJ 1738, feb ,1976.

ANS-75A ANSI/X3/SPARC (Standards Planning and Requirements Committee): *INTERIM REPORT from the Study Group on Data Base Management Systems*, FDT Bulletin of ACM, Vol 7,No 2, 1975.

BEN-76A Benci E., Boulet P., Bogaert H. and Cabanes A.: *Concepts for the design of a conceptual schema*, in (IFIP-76A), p. 379 ff.

BIL-76A Biller H. and Neuhold E. J.: *On the Semantics of Data Bases : The Semantics of Data Models*, Report (Institut für Informatik, Universitat Stuttgart, D-7000, Stuttgart,Herdweg 51),1976.

BRA-76A Bracchi G. Paolini P. and Pelagatti G.: *Binary Logical Associations in Data Modeling*, in (IFP-76A), p. 255 ff..

BUB-74A Bubenko jr J. A. and Berild S: *CADIS System 4: A tool for incremental description and analysis of systems*, Research Group CADIS,Univ. of Stockholm,Dept of Information Processing, Report TRITA-IBADB -3082,1974 (also presented at the BIFOA/GMD International Symposium,on *Organization Structure and the Structure of Information Systems*, GMD,St. Augustin, Fed. Republic of Germany, June 26-28, 1974)

BUB-76A Bubenko jr. J. A., Berild S., Lindencrone-Ohlin E. and Nachmens S., *From Information Requirements to DBTG- Data Structures*, ACM Sigplan/Sigmod Conference ,March 22-24,1976,Salt Lake City, Utah,USA.

BUB-76C Bubenko jr J.A.: *Information modeling methods: a survey and an approach toward a conceptual basis*, in preparation,to appear (1976).

CHN-75A Chen Peter Pin-Shan: *The Entity-Relationship Model -- Toward a Unified View of Data*. Report, Center for Information System Research,Sloan School of Management, M.I.T., Cambridge, Mass., 02139 (also presented in (RER-75A)).

CYL-62A CODASYL Development Committee: *An Information Algebra* , Communications of the ACM ,1962, pp. 190-204.

CYL-71A CODASYL System Committee : *Report on the CODASYL Data Base Task Group*, ACM ,April 1971.

COD-70A Codd E.F.: *A Relational Model of Data for Large Shared Data Banks*, Communications of the ACM, Vol 13,No 6 June 1970, p. 377 ff.

COD-73B Codd E. F.: *Relational Completeness of Data Base Sublanguages*, in (RST-73A), P. 65 ff.

CGR-73A Couger J.D.: *Evaluation of Business System Analysis Techniques*, ACM Computing Surveys, 1973.

CGR-74A Couger J.D. and Knapp R.W.: *System Analysis Techniques*, John Wiley/Sons, N.Y.,1974.

DAT-75A Date C.J.: *An Introduction to database systems*, Addison Wesley, Reading, Mass., 1975.

FAL-75A Falkenberg E.: *Design and application of a natural language oriented data base language*, Adv. Course on Data Base Languages and Natural Language Processing, Freudenstadt, Black Forest , Fed Republic of Germany, Aug , 1975.

FAL-76A Falkenberg E.: *A Uniform Approach to Data Base Management*, in (IFP-76A), p. 196 ff..

HAL-76A Hall P., Owlett J. and Todd S.: *Relations and Entities*, in (IFP-76A), p. 430 ff..

IFP-75A IFIP TC 2 SPECIAL WORKING CONFERENCE: *A technical in-depth evaluation of the DDL*, Namur, Belgium, PRE PRINTS, Jan. 15-17,1975.

IFP-76A IFIP-TC-2 Working Conference on *MODELING IN DATA BASE MANAGEMENT SYSTEMS* ,January 5-9, 1976,Freudenstadt (Black Forest),Fed.Republic of Germany (Preprints).

KAN-75A Kahn K.M.: *Mechanization of temporal knowledge*, Project MAC, Report MAC-TR-135, Sept., 1975.

KER-75A Kerr D.S. (ed.): *Proceedings of the International conference on VERY LARGE DATA BASES*, ACM,Framingham,Mass.,U.S.A.,Sept.,1975.

KIM-74A Kimble J.W. and Kallman K.L: *DATA BASE MANAGEMENT (Proceedings of the IFIP Working Conference,Cortina, 1974)*, North Holland/American Elsevier, 1974.

LAN-66A Langefors B.: *Theoretical Analysis of Information Systems*, Fourth ed., Studentlitteratur/Auerbach, Lund, Sweden,1973.

LAN-74C Langefors B.: *Theoretical Aspects of Information Systems for Management*, IFIP Congress 1974, pp. 937-945.

LAN-74E Langefors B.: *On Information Structure and Data Structure*, Univ of Stockholm,Dept of Adm Inf Processing,Report TRITA-IBADB-1019, Stockholm, 1974 (also in preprints of (IFP-75A))

LID-74A Lindgreen P.: *Basic operations on information as a basis for data base design*, IFIP 1974, pp. 993-997 (1974).

RST-73A Rustin R. (ed.): *DATA BASE SYSTEMS*, Prentice Hall,1973.

SEN-73A Senko M.E., Altman E.B., Astrahan M.M. and Fehder P.L.: *Data structures and accessing in data base systems*, IBM Systems Journal, No. 1, 1973, pp. 30-93.

SEN-75C Senko M. E.: *The DDL in the Context of a Multilevel Structured Description : DIAM II with FORAL*, in (IFP-75A), pp.269 - 295.

SEN-76B Senko M. P.: *DIAM II: The Binary Infological Level and its Database Language - FORAL.* ACM Sigplan/Sigmod Conference, Salt Lake City, Utah, March 22-24, 1976.

SUN-73A Sundgren B.: *An Infological Approach to Data Bases.* Urval 7, SCB(Statistiska Centralbyran), Stockholm, 1973.

SUN-74A Sundgren B.: *Conceptual Foundations of the Infological Approach to Data Bases,* in (KIM-74A), pp. 61-94.

YOU-58A Young J.W. and Kent H.K.: *Abstract Formulation of Data Processing Problems,* in (CGR-74A), pp. 259- 274.

AN ANALYSIS OF SOME CONSTRUCTS FOR CONCEPTUAL MODELS

N.A. Schmid
Institut für Informatik
Universität Stuttgart
Azenbergstrasse 12
D-7000 Stuttgart 1

This paper will try to clarify some issues that are of relevance for the discussion which model should be used in the conceptual scheme of a "three-schemata" data base management system.

Different alternative constructs, by which certain concepts can be represented in a conceptual schema, will be analysed and compared. Our opinion is that such a comparison should not be based upon personal beliefs and preferences. Therefore, the method, which will be used to make our analysis as precise and objective as possible, is to specify mappings between the constructs to be compared. These mappings indicate clearly in which way different constructs are related to each other, and what their particular advantages and disadvantages are.

Since many problems become evident only if we consider not only static structures, but also the operations to be executed on these structures, we will include "conceptual operations", i.e. operations that are to be executed on a conceptual schema, in our analysis.

The approach presented serves a twofold purpose. First, we use it to clarify some problems that are currently under discussion. Secondly, we believe that it could be applied also to other problems in the area to make the discussion more objective and precise.

1. INTRODUCTION

Lately, a considerable number of data models has been proposed for usage in data base management systems. The discussion, which of these data models would be suited better for the user, produced amongst valuable insights also many statements of personal beliefs and preferences.

According to the "three schemata approach" ([AS 75],[NI 76]), which seems to be accepted to some degree, every user may use the data model that he prefers. The question is now, which data model, information structure model, or conceptual model is suited best to be used in the conceptual schema.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

PHYSICAL OBJECTS, HUMAN DISCOURSE AND FORMAL SYSTEMS

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

19

Architecture and Models in Data Base Management Systems, G.M. Nijssen, (ed.).
North-Holland Publishing Company, 1977

PHYSICAL OBJECTS, HUMAN DISCOURSE AND FORMAL SYSTEMS

Ronald Stamper
London School of Economics
Houghton Street, London WC2

The central problems of database design, for the information analyst, are how to select the appropriate universe of discourse and how to relate the data structures, within the information system, to the real entities in the world. These are the problems of operational semantics and they are quite different from the formal semantic problems which arise in the study of data management. Their solution depends upon our being able to rely upon the stable norms of human discourse which are established when people use language and other signs for some practical purpose. A conceptual schema embodying the operational semantics of the system should not be confused with a general schema to contain a canonical data structure. The issues raised are illustrated by concentrating upon the simplest of semantic problems: the identifying of physical objects. The analysis presented arises from the LEGOL Project*.

Introduction - Information Analysis

By examining the simplest problem of information analysis we shall see that a database is not a model of reality but an embodiment of a myth. What could be simpler than the use of data elements to represent the physical objects, the tangible reality represented by a database? That is the problem we shall examine.

The information analyst specifies what data are required to solve some class of organisational problems. The designer of the information system must ensure that the data in the formal system are correctly linked to what he will call the 'real entities' in some 'object system'. They create an appropriate myth. The reality lies in the operational success of the system. If the system helps us effectively to solve some set of practical problems, then the myth upon which it is based is real enough. Change the problem, change the purpose of the system, and the myth may become inappropriate. A database, for all the logical precision of its structure rests on this quicksand. Only a continual analytical vigilance can keep it afloat.

Information analysis is being studied at the London School of Economics in a study of administrative systems based on complex rules. This, the LEGOL Project, uses statute law as experimental material. By attempting to devise a formalism which can express the kinds of rules that might appear in a statute defining a tax system, for example, it is hoped to discover a way of specifying an information system at a very general level. A second prototype interpreter for this language is now being designed, an essential part of which is a semantic model. The result of information analysis may be viewed as a semantic model for an application, possibly in some area of law. It appears that the semantic model of LEGOL is the

* The LEGOL Project is supported by the UK Scientific Research Council with assistance from IBM(UK) Scientific Centre, Peterlee.

same thing as the conceptual schema of database studies*. If this is so, then the LEGOL Project suggests that recent discussions of the conceptual schema have failed to recognise some important ideas and to distinguish two quite different lines of enquiry. The purpose of this paper is to explain these ideas and the method of enquiry being employed in the research.

Real World and Formal System

In the discussions of conceptual schemas there is evidence that two quite different problems have not been adequately separated. Fig. 1 is intended to emphasise some major features of our problem.

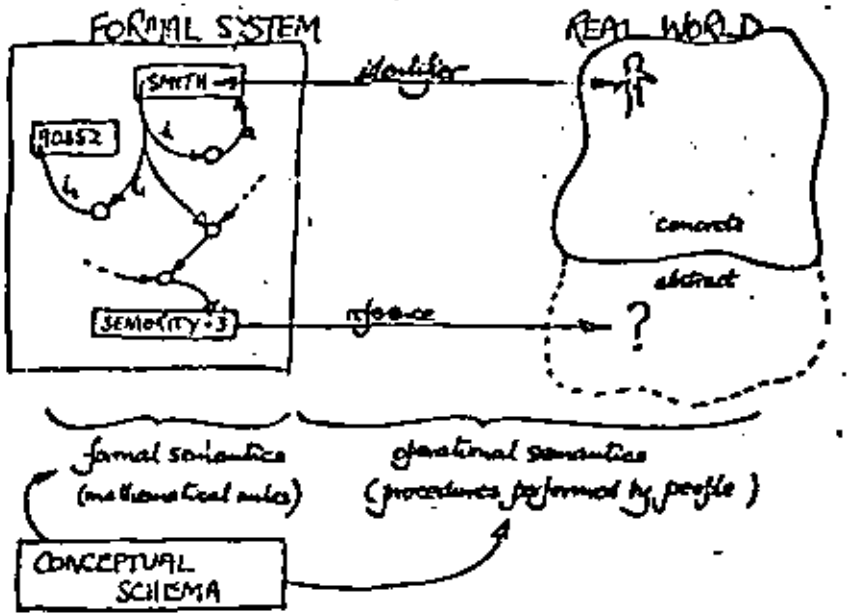


Fig1: Two aspects of semantics, formal + operational

The formal system contains strings of symbols - ink marks on paper or electromagnetic traces - and some of these may constitute an identifier of a person. The real world may contain the real person himself - you may meet him and shake him by the hand.

* This was not obvious to us on reading the ANSI-SPARC Interim Report (3) but this is the firm opinion of Dr. T.B. Steel, Chairman of this Committee, expressed at IFIP TC2 meetings in Pont-a-Mousson, September 1973, and Nice, January 1977

To a manager or administrator, who wants, say, to train his staff appropriately the identifier is the link to the person he can shake by the hand. It is easy to put into a database many millions of strings of symbols but the manager wants to know that the 'facts' they express are a reliable basis for action. To imagine that a database contains knowledge in this sense, merely because its structures of hierarchies, relations or networks can yield strings of printed symbols which may be construed as statements is naive. The manager's is an epistemological problem. No mathematical analysis can guarantee to the manager that the statements elicited from a database by his enquiries constitute 'real knowledge'. This can only be done by ensuring that there are operationally effective procedures for giving things names and for finding the real thing, given the name. By 'operationally effective' is meant that the procedure can be carried out, by the people who must use the identifier, with sufficient reliability to enable the information system to do a useful job*. The epistemological semantics of the database are embodied in these procedures which people perform. They may be difficult to establish and costly to support.

Mathematical methods, however, are appropriate to formal semantics. If we ask for the meaning of one string of symbols in terms of others, we need never leave the formal system in which mathematics can legitimately describe these relationships. In Fig. 1 the formal equivalence of 'SMITH' and the employee number '90352' is one such relationship. Other formal relationships, as the figure suggests, may link an abstract concept such as 'SENIORITY' to the more concrete concepts. These formal operations must be added to the operational procedures in order to establish the epistemological semantics of an abstract concept. This can be seen from the notion of 'SENIORITY' which is only connected to the concrete world via the formal procedures **. Hence formal semantics is a proper subset of epistemological semantics. The conceptual schema must therefore contain descriptions of both the internal, formal operations and the external procedures to establish the meanings of data.

Two Conceptual Schemas

Within Fig. 1 there is lurking a second, quite different problem. The formal system depends upon mapping the strings of symbols it contains upon suitable storage structures. This is as true of a clerical system as of a computer system but in the latter case it is crucial to the design of expensive general purpose software. This is where the second notion of a conceptual schema arises. It is a quest for a canonical structure to which all mappings of sub-sets of the data onto physical storage may be related. This view was exemplified in the statement by Nijssen (11) which placed the one conceptual schema between many external schemas (serving different sets of programs and many internal schemas (serving groups of physical groups of physical storage devices)). This kind of conceptual schema is naturally sought if one is preoccupied with the computer, the efficient use of hardware and the writing of programs. These problems are large enough to warrant the undivided attention of some researchers.

A serious mistake is made if we confuse these two notions:

CS(A) a schema of application concepts
CS(D) a canonical structure of data relationships

Both are essential but they serve quite different purposes and, further, they are related. Confusion, regrettably, is more common than a careful differentiation of CS(A) and CS(D). One reason is our habit of using mnemonic labels to talk about formal data elements. By discussing CS(D) in terms of examples where data elements are called EMPLOYEE NO., DEPARTMENT NO., and so on, we import into the discussion notions that belong to CS(A). With care we can avoid cheating by using our intuitive knowledge of the semantics of such elements but the ease with which we feel we can handle the epistemological problems of EMPLOYEE NO., DEPARTMENT NO, and so on leads us to suppress the problems of CS(A). The result is to imagine that a canonical structure of suitably labelled data elements will capture both the contents of CS(D) and those of CS(A) as well. This assumption is implicit in the majority of papers on database architecture (12, 13, 14). The view adopted here, however, is that the application concepts, whilst embodying structures that must be reflected in the canonical data structure, must embody a great deal more having nothing to do with the formal properties of the database.

To exemplify this, consider a description of a house. We want to know:

(a) Numbers of bedrooms and reception rooms
(b) Is it fit for human habitation?
(c) Is it free of land charges?

For the information system at large and the epistemological semantics of the data it is essential to know that

(a) is a relatively objective attribute which can be supplied by any normal person who goes to check,
(b) is a value judgement which must be made subject to certain constraints, by a qualified surveyor, whilst
(c) can be determined by computer search of Land Registry files where all such charges must be registered to be valid.

Such semantics are irrelevant to CS(D). For the designer of the information system these concepts are fundamental and they belong in the CS(A).

Following the information analysis phase and the specification of CS(A) a limited amount of information about the application must be passed to the designer of the computer system. As it relates to the data, this limited information may be lodged in CS(D). Essential questions we need to ask are: what information can be discarded as we move from CS(A) to CS(D)? and in CS(D) what canonical forms are appropriate for the data structures?

To see how these questions can be answered we must remind ourselves of the technical problem which CS(D) help us to solve: how to navigate through storage volumes to find the data elements wanted by programs. Obviously this entails naming the data elements so that they can be allocated to storage in any way appropriate to the storage topology (this allocation is contained in the internal schema). The names used in the CS(D) could be mnemonics but they need not be. Let us suppose that they are not, so that there is no temptation to transfer subconsciously or covertly any knowledge of the epistemological semantics. What then do we understand about the external schema which maps the data names used in applicatio programs into the canonical form of the CS(D)? A CS(D) full of data names without self-evident meanings will not help the application programmer to set up a mapping of his data elements into those of the canonical data model. He will be forced back to CS(A) where he may find that the meanings of data already in the database are not appropriate to his new task.

---

* Notice how this statement emphasises the importance of knowing the purpose of the system, a recurrent theme in this paper.

** Abstract concepts e.g. 'GUILTY' may not be entirely formal but, to a greater or lesser extent, the embodiment of someone's value judgement or prescription. This is dealt with in general terms in (5) and briefly in the context of LEGOL in (1). Space does not permit the exploration of this topic in this paper.

not only names of data elements but relationships among these elements must be contained in CS(D). To investigate the signification of these relationships, we can strip the problem bare by not permitting mnemonics to be carried across from CS(A) to CS(D). If we do this, it is clear that 'logical' notions such as 'entity', 'attribute', 'rule' and so on, cannot be used as in CS(A) when talking of real things. Our database software is intended to help us 'navigate' through storage to the data we want. The CS(D) is the 'chartroom'. 'Logical' relationships are significant because they define routes among data elements that must be available and may be used quite often. From this point-of-view, the dispute between the devotees of binary and n-ary models falls into perspective. If the computer permits only sequential processing then one must travel between single data elements and the binary model is natural basis for the canonical form such as Senko describes in (15). If the computer is more appropriately used as a parallel processor, then groupings of elements which are necessarily and only necessarily related will be important structures to identify. This leads to Codd's third normal form. Something is gained and something is lost which ever canonical form is used. The psychological discomfort of the binary model is pointed out by Senko in his paper. The artificiality of the binary model is an advantage if you wish to avoid confusing CS(A) and CS(D). The relational view of data is convenient but dangerous at the CS(D) level but at the CS(A) level relational structures seem to be essential. At least this is strongly suggested by our studies of information analysis for legal systems.

Having attempted in this section to draw a demarcation line between application concepts, CS(A) and canonical data structures, CS(D) we shall turn to the CS(A) without the risk of becoming involved simultaneously in two sets of difficult and complex problems. We introduce these ideas as they arise in the LEGOL Project.

The concrete and the abstract worlds

The experimental material used by the LEGOL Project is legislation of the kind upon which routine administrative systems are based. These systems of rules are constructed so that they are grounded, as far as possible, upon simple, reliable concepts such as physical objects, events and relationships (see Fig. 2) whilst the abstract notions such as 'a work of literature', 'copyright', 'ownership of copyright' and so on, are precisely explicated through formal rules.

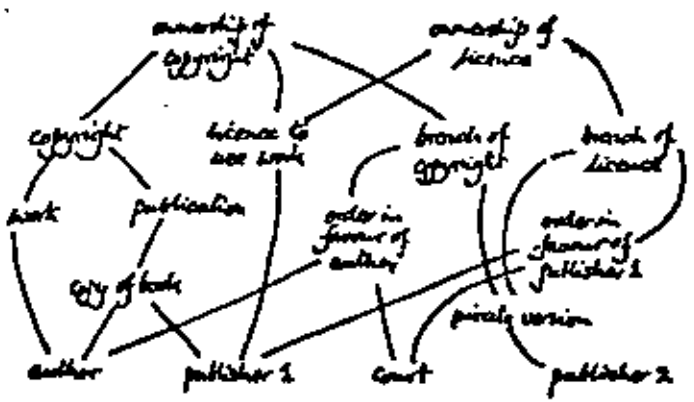


Fig 2: Abstract entities grounded upon physical ones

LEGOL is intended to describe such a structure. Both the analysis of concepts and the expression of legal prescriptions is included and both belong, strictly speaking, to the conceptual schema (by which is meant CS(A) in the rest of this paper). One may imagine that the legal draftsman is trying to write a program to govern the behaviour, not of a computer, but of the real world. In a sense, the LEGOL version of the rules and their concepts have the same function and, therefore, by the earlier analysis, the conceptual schema has this role of 'programming' the real world. Fig. 3 on the left hand side shows this.

Fig 3: Two functions of LEGOL as conceptual model

Simultaneously, the LEGOL version serves as a computer program because there is a computer system* which will interpret it. This is represented on the right hand side of Fig. 3. The LEGOL Project has effectively escaped the extra complications with which CS(D) must deal by accepting the solutions built into the software employed by the prototype, namely PRTV and MP/3 (16 and 17).

From this analysis, it would seem that a conceptual schema, expressed by LEGOL or in some other way, would deal fully with the link between formal system and real world. This is not so. We are also dependent upon the ways in which people understand the words, numbers, sentences and so on, which cross the formal system's boundary. This neglected topic is the one we shall now examine.

Discourse Systems

Formal information systems are not products of nature. They inhabit the system of natural discourse which human beings have created as their most important 'tool'. Natural language serves as the meta-language in which formal systems are defined. This discourse system complicates the picture which was described above but it also provides the essential means of escape from the infinite regress of futile logical analysis.

* albeit a crude, limited prototype at present

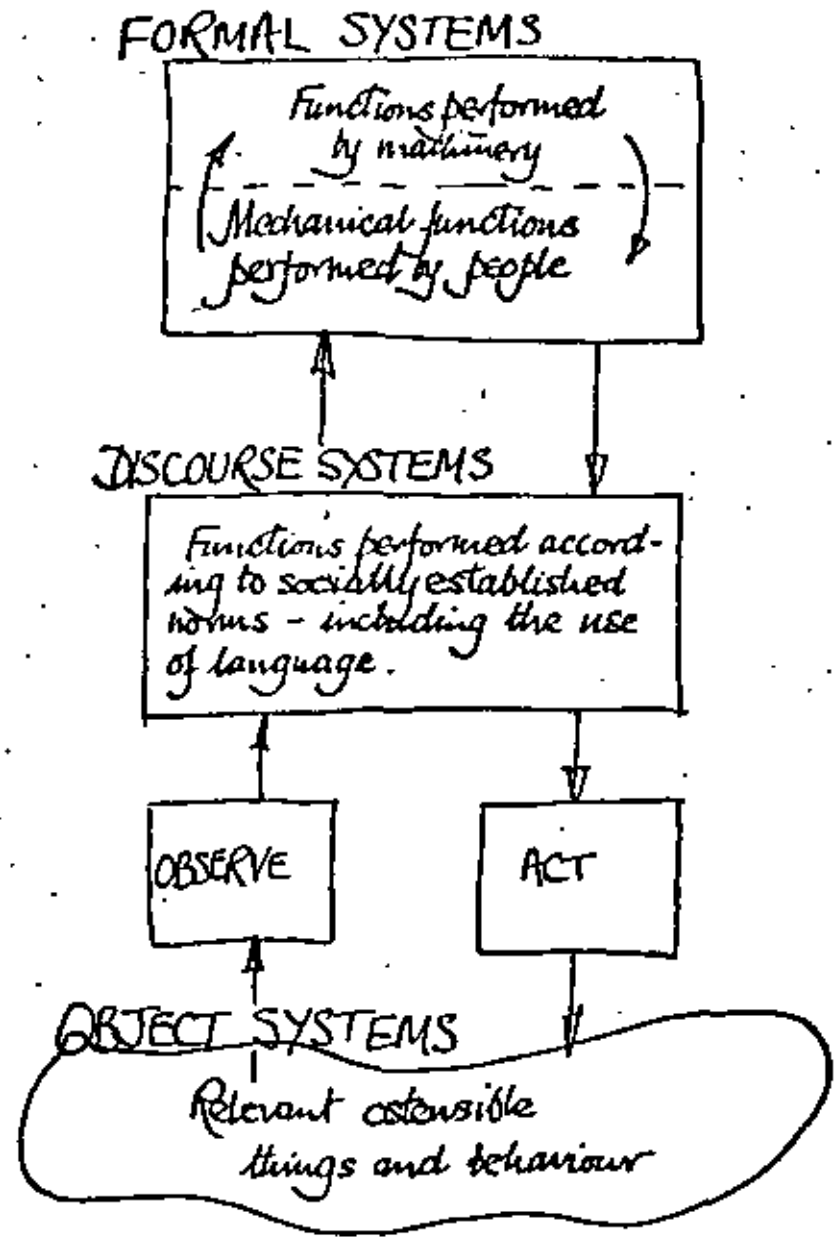


Fig 4: The intermediate role of the discourse systems

The symbols in the formal system are related to the things in the object-system by people using natural language. See Fig. 4. It is convenient to call the system in which the meanings of signs depend upon linguistic and other norms, the discourse system*(4). It corresponds to the informal part of an organisational information system. It evolves through human, socio-linguistic intercourse, unlike the formal system which depends upon explicit definitions. When observations are made, resulting in data-values being inserted in a database, or when datasets retrieved, as a basis for action, the link will be made by words or other signs which are meaningful both in the formal system and the discourse system; e.g. 'CUSTOMER, J. SMITH LTD'. When symbols of the formal system are used without their having a place in the discourse system (e.g. an account number '3708J455') their use must be explained to the relevant people, in natural discourse*.

Operationally, the symbols we call 'identifiers' will be linked to the relevant physical objects through the discourse system. The quest for logical certainty in this link is a blind alley. We shall always be dependent upon the stability of human norms in the use of signs. These constantly shifting norms are the slightly insecure but only available foundation upon which a practical database can be built. Surprisingly, despite its central role in explaining any large data-processing system, the concept of a 'discourse system' has been overlooked by information system theorists.

Even when dealing with the apparently simple semantic problem of how to identify physical objects, the information analyst cannot rely upon the norms of the discourse system. The way in which the real world is partitioned into objects depends upon the use of language within the discourse system, when people are solving practical problems. It is a mistake to assume that there is a unique, objectively-given set of physical objects to which language refers. The spatial and temporal limits of the objects we talk about are a function of why we are referring to them, of the shared problem we are trying to solve through the use of verbal and other symbols. If the purpose is changed, so are the objects. The norms of the discourse system can shift in subtle ways to match the shifting context of different problems.

This thesis needs to be demonstrated and this is done below. If it is true then it implies that formal systems call for the use of words in stable and uniform ways that differ from the usage familiar in everyday discourse. It will be a task of the conceptual schema to embody these special meanings of data in the formal system and to help users to employ them correctly.

Entity and Purpose

Presumably we want to build formal information systems which enable people to cooperate and do useful things in the real world. The basis of human cooperation is the use of signs, including words and other symbols (5), so that one person may observe and others may know or one person may instruct and others may act. In any use of information there will always be a closed loop from observation, through decision, back to action affecting the thing observed. In an informal use of words to solve a practical problem, very few people will be involved in this cycle and there will be no long delay between observation and action. Formal systems tend to supersede informal discourse when many people have to cooperate in a complex task which requires precise action over long distances and extended time periods. Formal systems are the essence of many kinds of organisation.

---

* Note that discourse involves more than natural language; it depends upon learning how to use words, numbers or other signs to get things done.

The operations of observing and acting are the interfaces between the object system and the information system. The computer specialist is familiar with the problems of organising and labelling symbolic structures within the formal information system. In this paper, we are concerned with the quite different problem of how to organise and label the contents of the object system*. The semantic problem may be regarded as that of establishing correspondence between entities and data. Just as we distinguish and label data to navigate through storage space within the computer, so do we distinguish and label entities to navigate through real space and time. Just as we wish to update a record consistently and later retrieve it, so do we wish our observations and acts in the real world to be consistent. For example, if traffic warden A observes a vehicle obstructing the road, then that vehicle and no other should be towed away by constable B and its owner and no one else should be fined. Any non-computer specialist would be amazed that the identifier problem should be presented this way round, but discussions of identifiers by data-processing specialists are almost invariably in terms of locating records in files and dealing with them consistently. In the literature, we find that discussions, ostensibly about identifying entities, usually turn out to be about identifying and locating records in files: the information retrieval problem.

The semantic problem is much more difficult than the information retrieval problem, despite their formal similarities. The fundamental difficulty about identifying things in the real world is that there is always room for dispute. Testing for sets of matching symbols in storage is relatively simple. Identifying some entity in the real world has no absolute outcome. We must aim to provide a person with an identifier which enables him to find the entity with sufficient precision for some specific purpose, such as levying tax. We must be prepared to redefine the entities when we change our purpose, for example, 'factory' for tax purposes, would be differently interpreted from 'factory' in the context of industrial safety legislation. In computing terminology, one might say that for entities there are many 'subschemas' but no general 'schema'.

The danger of supposing that one, universal, entity-structure can be imposed upon the real world is illustrated by the philosophical problems that this supposition will generate. By abandoning the belief in a universally valid picture of the world in favour of a series of different pictures, each serving some practical purpose, these philosophical problems evaporate (6). The effect of purpose on the entity-structure is copiously illustrated later, in the section on entity-boundaries, meanwhile it may be enough to consider the classical problem of Heraclitus's river. One cannot bathe twice in same river, though, in a sense, it is still there. This paradox is like the confusion of two databases, one to be used by bridge builders and one for environmentalists.

The same, dangerous supposition of a universal entity-structure is embodied in a database management system which requires a single, general schema to be established. A mathematical or computer-centred approach to database design tends to encourage the view that the world has a self-evident, unambiguous structure which can be recorded in files. The distinction made earlier between CS(A), application concepts, and CS(D), the canonical data structure, is basic to solving the problems of database semantics. We must constantly remind ourselves that our discussions about 'customer-records' and 'product records' and 'stock-records' usually have nothing to do with the real world of customers, products and stocks. These terms refer to groups of symbols to which we should attach codes such as "7364', '8421' and '3020', instead of 'CUSTOMER', 'PRODUCT', and 'STOCK' with their extraneous

* The physical objects which themselves carry information (e.g. documents) are themselves part of the object-system. In so far as we are interested in them as objects rather than signs, they may be treated in the same way.

and strictly irrelevant connotations. The symptom of this kind of error is the use of a term, such as 'customer', without qualification. In a real company, 'customer' will have many meanings to suit many users - lawyer, accountant, salesman, market researcher, production controller. The analysis of the entities referred to in legislation is made this clear because there are frequent shifts in meaning which the draftsman must make explicit. The LEGOL system, therefore, requires every entity definition to be given a context. This is a difficult problem for the designer of database software but it is better that we face up to it. I conjecture that a monolithic database can easily become an embarrassing organisational white-elephant.

One task for the semantic model is to keep track of the varied purposes for which data are used. This information is typical of that used initially by the system-designer and then forgotten once it is only implicit in the final physical design. It should be held in CS(A) and discarded for the narrower purposes of CS(D).

Entity and Stability

Our ability to use computers correctly within an organisational system depends upon our ability to use signs, such as words, numbers and various codes, in a stable way so that the observer of say, a faulty component, can make it in a production report knowing that a colleague in another factory will be able to identify it and repair it. Many people may observe and work upon the same component in ways that are coordinated through the exchange of signs. The component may change location, colour, texture, shape, function and so on yet, through this flux, the concept of that component will be sustained by a group of people who are able to identify it with a constant name. When Quine refers to 'the myth of physical objects as a device for working a manageable structure into the flux of experience', (7) to some people he may seem to be adopting a quaint philosophical stance, but not to anyone who has thought carefully about the problems of designing a database for a production system. Materials change their shape and appearance; batches form, merge and disperse; assemblies are created and their components changed; throughout this flux a limited but effective entity-structure must be used; the formal system could not economically recognise every possible discernible physical object.

The effectiveness of any information-system depends upon stable patterns of human verbal or symbolic behaviour known as 'perceptual norms'. These norms belong to the discourse-system. They come into existence because people share problems which they solve cooperatively through the use of language. Given the context and the problem, a group of people will evolve a way of attaching words to reality so that common perception of the problem can be obtained; failure on the part of an individual to perceive the problem in the same way as the majority will result in his acting inappropriately; criticism from his colleagues will lead him to adjust his perceptions into line with the established norm. Through this social mechanism (and not through the application of explicit definitions) words are assigned to things in a stable way and entity-structures are imposed upon the world about us. These structures of perceptual norms may be sustained through social interaction. They may lead to poor solutions to practical problems, particularly when the interaction is purely verbal or symbolic; norms are exposed to objective criticism only when they give rise to events in the real world of physical objects and people. For example, the impounding of the wrong car and the charging of the innocent owner for obstructing the highway would suggest that the symbolic 'image' of the real world used by the police in the files of the road-licence and traffic-offences system is 'out of focus'. Many information systems interpose abstract concepts between observation and action. These, such as the legal object 'company' or legal relationship 'ownership' help us to create a stable picture of a changing world, but their meanings are ultimately dependent upon the semantics of physical objects and events. Normally it is easier to resolve disputes about physical things than disputes about other entities. This

justifies for example a careful review of how we attach names to physical objects as a prelude to the study of more tenuous concepts. This example of a semantic problem is about the simplest we could choose.

The functions of names

Dewey (8) suggests that a name serves three functions: as a fence, as a label, and as a vehicle. We partition the world into physical objects by naming them; changes may take place within the boundaries but we keep the fences intact. This partitioning of the world provides us with a reference-frame, a kind of generalised coordinate system by which we can navigate. Names are the coordinate labels. We attach a label to any point to which we ourselves might wish to return or to which we might wish to direct someone else. Once the object has been given a name, we can transport it, as it were, symbolically, and place it in new juxtapositions with other objects, without the need to shift the object physically. Names, as vehicles, are the means by which we discover, and impose upon the world, new and perhaps preferable arrangements by means of our conjectures, hypotheses, theories and plans. Let us examine each of these functions more closely from an information analyst's point-of-view. Information systems depend for their effectiveness upon the correct assignment of names to things, the appropriate manipulation of those names followed by the correct association of things with the names.

Let us first observe that the information analyst is concerned with the definition of the formal information system in which the meanings of words, numbers and codes are formally defined. The meta-language upon which his definitions depend is the natural language of the discourse system which, in turn, is rooted in the perceptual norms arising from social interaction.

The fencing off is done by the discourse system. Normally, we employ two or more stages, beginning with the recognition of an entity-class before we name the [illegible] the ordinary use of words is generally adequate for [illegible], the normal usage is not precise enough; we then have recourse to definitions (as in the Copyright Act 1965 c73 where, for example, "'written matter' includes any writing, sign or visible representation".) The first step is intended to provide operationally effective procedures for differentiating the physical object from its surrounding. It may involve specifying who should make the observation in those cases where the special skills of, say, a physician or valuer or tax inspector, may be required. The instruments to be used, the place of observation and other circumstances may have to be included in the definition. Thus the physical object is identified initially.

Having found* the object, we must attach a label to it. In the everyday situations dealt with by our discourse systems this is simple because the number of objects we are concerned with is small. In a formal information system employing a computer, the number of occurrences may run into millions. The construction of these labels for use in an information system is complicated in ways that will be discussed below. The identifier must not only correspond mathematically with the unique physical object it represents, but it must correspond operationally. The information system will not be able to do useful work unless there are procedures for locating and for checking the identity of the object specified by any identifier. (Consider, by way of illustration, the operational problems of locating an object in a large museum or of associating code-numbers with billets of red-hot steel) These are central issues in the design of codes for identifiers, a topic discussed further below.

---

* Objects that are too large or too small to present themselves for observation in a simple way (e.g. the solar-system or a molecule) must be treated as abstractions. See Popper (9) on observationally-testable statements.

Having established an operationally effective, one-to-one relationship between physical objects and identifiers, we can begin to use these carefully constructed signs as vehicles. But we shall need 'traffic' rules. By re-arranging the signs which represent the objects in the world, we conjecture new arrangements of the world itself (for example, orders sorted by product type may be assigned to appropriate machines). New arrangements conjectured in this way will only be translatable into reality if the identifiers have not been moved beyond the bounds within which they are meaningful. The information analyst must establish these boundaries if the data-processing system is to be constrained to produce meaningful outputs. (Thus: engine 1234 cannot be used simultaneously in two distinct cars but an abstract object such as the play, 'Hamlet', may be used simultaneously by several theatres) This is the central issue in the semantic problem of updating files which is touched later. Each of the functions of a name will now be examined in more detail.

Entity-boundaries in a formal system

A formal system must rely on a given entity structure and normally it relies on the discourse system to assign the names it uses. Reliance on the discourse system is ultimately unavoidable but there may be a converse influence of formal rules on the the norms of discourse when everyday terminology needs to be tightened; rules may help to make the naming structure more widely consistent than it would be under an unaided discourse system. For example, in the 1965 Rent Act, "'the occupier', in relation to any premises, means any person lawfully residing in the premises or part of them at the termination of the former tenancy"; this restriction removes ambiguity from the word 'occupier' as it is employed in normal discourse. Another reason for formalising the naming of things, common in technical situations, is the lack of any established perceptual norms upon which to rely; for example, a particular subassembly for an engine may never have been made before so what it constitutes must be defined, ad hoc, by an engineer. His definition will ultimately rest upon normal usage of words in the discourse system and any formal definition will be anchored similarly in the perceptual norms, deriving what stability it has from these cultural reference points, not from the logic of the defining rules themselves.

The information analyst has a responsibility for partitioning the world appropriately into its relevant component entities so that data about them may be manipulated meaningfully and consistently by the formal system. Let us look at some of the problems he might encounter.

Changes in the properties of putative entities may transform them beyond easy recognition and raise problems of when the entity ceased to exist, being transformed into another entity, perhaps. For example, since the advent of transplant surgery, the formal definition of the moment of death has become a matter of keen interest. In a manufacturing situation one can imagine a piece of steel being worked into many varied shapes, its metallurgical properties and its physical continuity being the only aids to its identification. The piece of steel, through all its transitions, will be regarded as the same entity, the reason being that we do not change our intentions about what to do with the steel as a result of the transformations it undergoes. The human body will be used very differently alive and dead. In general, the partitioning of the world into its physical entities will depend upon what we intend to do with the entities so categorised. Transplant surgery has made semantics a very lively subject!

Assembly-subassembly relationships offer innumerable alternative entity structures from which one must be selected, within some simple constraints. The components of each assembly must be mutually exclusive, otherwise instructions for making and dismantling are likely to be ambiguous; also the list of components must be exhaustive, otherwise the formal system will not be able to describe the manufacturing process completely.

The detail into which the assembly is decomposed will be limited by the need to act upon components individually, to manufacture them, order them, or redesign them. All these activities may share the same entity-structure but other structures are possible. For example, subassemblies may be distinguished on the grounds of the functions they perform. This criterion will suit the design and maintenance engineers who must understand the working of the whole assembly. It may be impossible to select one decomposition that will serve all purposes. The design engineer may prefer a decomposition based upon function whilst the production engineer may be more concerned with physical structure; maintenance may require elements of both, function to aid diagnosis, physical structure to aid repair. The problems of how to name the subassemblies in a complex structure obtrude even more severely when the main assembly is an organisation and the decomposition is an accounting framework. This difficult problem, aggregation/disaggregation or assembly/subassembly, may provide a major justification for using a computer in certain applications; the computer may permit information based on many different structures to be consistently related to a single, very detailed underlying entity-structure but the solution may be uneconomic if this common structure is too detailed.

Systemic continuity may be sufficient reason for naming, as a physical entity, something with changing physical composition. The proverbial example might again be the river that Heraclitus pointed out would not be the same one next time you stepped into it; the human body is much like the river in this respect but, more prosaically, there are, for example, production lines which, through maintenance and development, are physically entirely replaced whilst retaining their systemic identity. Physical objects are not always physically unchanging. Their permanence arises not from what they are made of, but from the constancy of the problems they pose and the ways we wish to use them.

Change or use or role may signal the end of the existence of one object and the beginning of another. Natural discourse would tend to acknowledge continuity in cases where a formal system would have to make more exact distinctions. For example, riding a bicycle up to the traffic lights leads to my son stopping, in conformity with the law relating to vehicles; but carrying the bicycle home to him for Christmas, I was carrying a parcel, so I was governed by different laws. Similarly, many a bottle has become an offensive weapon. This kind of shift can be treated semantically as the reclassifying of the same particular object under various generic names. Such an explanation will do for the discourse system, which has a restricted span of attention, as it were, the object may only be relevant when its role has been decided: the traffic law did not apply to my son's bicycle until Christmas morning when my parcel became his vehicle.

Different objects may be made from the same material presenting a problem similar to the one above. A sword may be turned into a ploughshare. We might record this event as the end of the sword's existence and the beginning of a ploughshare. The transformation may be quite a subtle one, as when an artist selects a shapely piece of wood as an objet-trouvé; the act of selection would bring this natural object within the terms of copyright law which would treat it as any other sculpture more elaborately fashioned.

From the above analysis, it must be concluded that our names for things do not embody any knowledge of the intrinsic nature of the world, rather do they reflect the nature of our problems, our purposes and our linguistic means of interacting with the physical world. If we are on such shifting ground when considering the semantics of physical objects, how much more cautious we shall need to be when we come to consider more abstract entities and properties! At least we should be prepared to look closely at the glib assumptions made about shared databases. Data obtained for different purposes are likely to be semantically different. They should be shared only with the utmost caution. For example, 'child' seems a simple notion but in data-processing systems for family allowances and for medical record linkage the definitions would be difficult to reconcile.

For the purpose of the LEGOL Project, the above analysis gives some useful direction to our treatment of semantics. If we are to define data groupings which are irreducible ones from a logical point-of-view, then it appears that we should make them correspond to physical entities, containing information which will be constant throughout its existence. For a physical object, the minimum set of elements is a name and times for start and end of existence. The invariance of the logical data-structure seems an appropriate way of reflecting the semantic invariance of the real entity-structure to which we are referring. All the time-varying properties would be treated separately. Surprisingly, it may be more appropriate to include purpose as one of the properties of an entity class. This conjecture raises the difficulty of defining purpose appropriately; in LEGOL, the nearest we come to doing this is to define each entity-class within a limited context such as a branch of law, an Act, a section or even a subsection.

The formation and assignment of names

In the discourse system, the giving of names is usually quite simple. The number of objects involved may be quite small and the names may be assigned temporarily for solving a limited problem. People supply their own names to avoid confusion; things are labelled by demonstrative pronouns, genitive nouns, possessive pronouns or by demonstrative or possessive adjectives plus a common noun. (Common nouns, e.g. 'book' are labels for entity-classes) Seldom in normal discourse do we need to attach proper names to large numbers of objects.

The naming of large numbers of objects is, by contrast, a major problem in formal information systems. Indeed, the reason for having a formal system is often the need to keep track of multitudes of things in a manner that is consistent over distance and time. Licencing and registration systems are typical examples. Normal discourse is adapted for more localized use. Before any object can become a part of the subject-matter 'known' to a formal system, it must be given a name in a way that satisfies some simple criteria. The names may also be chosen to reflect real structural patterns. These are the problems of forming names.

Each name must be unique. This can be achieved in various ways, the commonest of which is to assign names from a sequential list, keeping note of the last values used, e.g. accession numbers to library books. Names might be generated in more elaborate ways from names of related objects; for example, subassembly names as elaborations of their parent assembly names. Uniqueness may be guaranteed by the rules so that identical names will only be generated for identical entities. If this is not the case, the formal system must include a register of names, some means of checking that new names are unique and rules for modifying any homonyms. For example, a person's name may be enough to identify him in a personnel system, but provision may have to be made for the occasion when a new employee's name is already used by the system. Uniqueness is the only necessary characteristic of the name that is chosen for an object.

Structural properties of names are sometimes useful, but they are not necessary; sometimes they can be a source of dangerous problems. For example, plots of land might be named according to their parish, within local government area, and serially numbered within parish; this structure, reflecting relationships among the plots, would be useful until a change of local government boundaries. Accession numbers of books will indicate roughly when the library acquired them. More elaborate structures can be represented if a suitable isomorphic name-structure can be found.

Reassignment of names may be forced upon a formal system. Within one and the same system, structured names may have to be reassigned if the underlying, real structure is no longer correctly represented by the name structure. The local government reorganisation would require such effort, in the example cited above. The merging of two formal systems performing the same functions for different

populations of objects will require reassignment of names if there is any risk of the same name being used for objects previously dealt with by separate systems. This reassignment can be an entirely formal operation. But this would not be so when merging systems which deal with overlapping populations. Such a situation would call for procedures to check any likely synonyms resulting from the merger; these procedures may be expensive and slow, requiring positive physical identification of the objects and reference to both information systems. Problems would be minimised if naming always followed standards based upon underlying, stable structures; for example, using grid references for naming plots instead of the method suggested above. Standard rules for forming names would not, of course, solve the problem if the entity boundaries were drawn differently in the system being merged.

Identification of objects

Whereas the assignment of a name is an operation performed only on the accession of an object into the formal system's universe of reference, identification is the continuing use of the name to link the real object and the symbols that describe it. The link is two-way. For the formal information system to serve any organisational purpose, it must be possible for users to move reliably from object to symbol and from symbol to object.

The name of the object, to be an identifier, must be supplemented by the name of the entity class of which it is an instance. Often enough, the name itself is formed in a way that indicates its entity class (one may have an employee named 'FRED'; (yet women's names have been taken from flowers and given to furnaces!)) The intrinsic identification of class from the form of a name will be a useful local device, say, for handling some limited streams of messages from which the chance of ambiguity can be excluded, but in general, we have to assume that the name gives no clue to the entity class.

The first step from object to symbol is to distinguish and name the entity class and subclass down to the point in the relevant hierarchy of classes beyond which individual unique names are given. This can often be done by relying upon the familiar norms of the discourse system, if not, the information analyst will need to ensure that the relevant personnel are trained or advised how to apply the special classification rules which distinguish an entity from its surroundings and establish which naming procedure is applicable. The second step is to derive the name from the object. Often this is simple, you read the number printed on the document or on the ring round a bird's leg, for example. Less reliably you may take a number from a tag attached to a garment; less reliably because there are high risks of the tag having been removed and reattached incorrectly, or incorrectly attached in the first instance. As in the case of a credit card, one may use additional information such as a signature, to reduce the risk of incorrect use of a name; more information is used on a passport which includes a description and photograph. In some cases, the name cannot be 'tied' to the object, perhaps because it is too small (insect), inaccessible (star), a liquid (batch of dye) too hot (metal) or likely to be embarrassed (spy). In such cases, the observation of the object must supply enough information to enable one to construct its name. The same problem arises when an extrinsic label becomes detached from its owner. Systems are sometimes disastrously designed with insufficient regard for this problem.

The problem just cited is akin to the problem of going from symbol to object. Used in this way, we see identifiers as generalised coordinates for locating things in the world. If the identifier does not convey appropriate topographical information to the discourse system, as in general it will not, then the name serves only as a check that one has arrived at the correct point. Locating the object would, in general, call for the scanning of possibly all the objects of a particular type referred to by the system. Objects are usually found more easily by applying various established techniques. In the simplest case, the formal system relies upon the discourse system in which the people expected to use the identifier can be presumed to know the location of the object. Sometimes this knowledge is not readily available and the formal system must keep track of persons who may be expected to know (e.g. next of kin, owner). To establish the formal name of a person correctly, the system may have to be able to conduct a dialogue based on information obtained during earlier transactions with him, e.g. asking him to supply his mother's maiden name, a technique which is not, incidentally, universally applicable. In other cases, the formal system may keep track of the actual location of the object, perhaps noting a person's changes of address every year, or even by following movements second by second, as in an air-traffic control system. Sometimes, movements can be tracked if certain decisions about movements and consequent actions are recorded; in this category are batches of chemicals in a system of tanks or billets of steel rushing through a mill into specified locations in the cooling-beds. The problem, mentioned in the previous paragraph, of how to go from object to identifier when it can carry no label, cannot readily be solved unless the physical location of the object can be treated as one of its known identifying properties. There may be other, time-dependent properties necessary for identification, such as stage of manufacture; they must also then be logged by the formal system. Either the logging of these characteristics will have to take place literally continuously or the formal system will have to employ certain rules of continuity that will enable it, with sufficient accuracy, to extrapolate, from suitable, intermittant observations, to the current locations and states of the objects in its universe. These continuity rules are an essential part of the semantics of these objects. In a rigorous analysis they need to be made explicit. If there is any significant risk of the tag being lost from an explicitly labelled object then, to remain effective, the system must employ similar, but less exacting, continuity rules. Detecting errors in identifiers would be impossible without them.

There can never be one hundred per cent certainty that labels remain attached to the objects or that continuity rules are wholly reliable, hence the risk of identification error cannot entirely be eliminated. This risk has two components: finding the wrong object given its name and associating data about a known object with the wrong identifier. These are two important design parameters for the information system. The formal system must have checks built into it to ensure the desired levels of semantic reliability.

Incidentally, it should be noted that the case of the object which cannot be explicitly labelled serves to illustrate the idea that a name is an abbreviation for a detailed description of the object, detailed enough for the object to be selected uniquely among others of its type. The name might actually be formulated by encoding one such standardised description. See (10). Rules which, in a formal system, enable one to convert one identifying description into another, probably via the name, might be regarded, along with rules of continuity, as a means of strengthening the links between object and symbol. They too will have to be made explicit for use by a formal system. A suitable name might be rules of synonymy.

Rules of continuity and rules of synonymy are used when updating files to check for inconsistencies. These are examples of the traffic rules for Dewey's names as 'vehicles'.

Application in LEGOL

The foregoing analysis is incorporated in the semantic model of the LEGOL system where the idea of an identifier is not to be confused with the notion of a 'key' for information-retrieval. The LEGOL formalism is intended for expressing rules about the conduct of a business or a society. An identifier enables one to find the real objects to which the information system refers, a process dependent upon the discourse system.

structurally, the semantic model is surmounted by a number of entity-types which embody the rules for identifiers. The simplest type is the 'object' and the physical object the prototype for those abstractions which we treat conceptually as objects and identify, symbolically, in ways analogous with the identifying of physical objects. Characteristic of an object is that a name, unique to the individual, is a sufficient identifier. By way of contrast, we may consider the relation, another type, which links other entities. The prototype is the physical relation among physical objects. To identify a relation, we must know which objects are related, when and in what manner, such as 'on', 'under', 'inside'. A relation is not uniquely identifiable unless the period of its existence is known. So, for a relation we have the following identifier elements:

Relation: TYPE, OBJECT 1, ... OBJECT n, START, END.

In the case of an object, we need only the unique name as an identifier, but an object in the semantic model also incorporates the period of the object's existence:

Object: NAME, START, END.

These basic templates can then be used to define some of the 'traffic rules' of the semantic model. For example, the relation cannot exist outside the period of existence of any of the objects involved. A knowledge of when an object starts and ends its existence is also essential for its semantics to be precisely defined. Unless the three elements of an object are precisely established, it will normally prove impossible to build a formal information system to exercise control over those objects. For example, an analysis of the Family Allowance Act 1965 failed to reveal definition of the period of a family's existence.

Object names are assumed to be unique for all time. This can always be realised by suitably adapting a naming system which does not provide unique names. In this, 'unique' need only be interpreted within a context of a given problem. If that context is a dispute among three individuals, the naming of those parties raises no difficulty. When large numbers of individuals are involved, there may be formal rules for registration which govern the assignment of names. If so, the LECOL analysis could include them. If not, the semantic analysis will include sufficient information to enable the discourse system to act as a reliable interface between the object system and the formal system. Information of this kind is essential for building an application though it has no place in the programmer's view of a database system.

Before the assignment of individual names, the object class must be specified with sufficient precision. In the LECOL system, this is done by defining a hierarchy of 'entity-categories', each of them limited to a specified context or problem. Thus, an instance of an object class may be 'building' but this entity-category will be given different meanings in the contexts of revenue law and planning law. Although the semantic models in these two contexts may be almost identical, it will always be clear that the data-values have different meanings.

At the basic level of analysis, the person who is formulating rules in the LECOL language may give a particular interpretation to an entity-category in an appropriate context. At this point, he must be responsible for establishing the mechanics of the naming process.

Summary

This is the first of a series of papers about the LECOL semantic model which treats the concepts of entity and identification from a point-of-view of systems analysis rather than as a problem of information retrieval. This leads to a conceptual schema more complex than envisaged in previous discussions of DBTG. The added complexities make explicit characteristics of object systems and discourse systems which are later submerged within the design of a particular physical information system.

Only the narrow topic of how to link physical objects to the signs or symbols that identify them has been considered in this paper. Philosophical problems are avoided in system design because we use symbols only with sufficient precision for a specific purpose. The fact that philosophical questions bedevil any attempt to use language in universal ways, suggests that when we extend the scope of a formal system, we must check that the semantic assumptions upon which it is based can cope with its extension.

At least, philosophy has helped us to realise, contrary to popular opinion, that the world is not composed of so many well-defined objects ready to be named but rather, that it is structured into entities through the agency of language. An entity-structure depends for its stability upon the conceptual norms established and maintained by natural discourse in a social system. The choice of norm is a reflection of the problem shared by the people who sustain the relevant norms. To change their problems is to change their perception of reality and their use of words.

Formal systems can only use signs in ways that are rooted in the norms of the discourse system. Physical objects have the simplest kinds of identifiers: just a class of entity and a name. We must start by clarifying, for a specific formal system, how the physical objects it is to recognise are to be fenced off from one-another. Many examples were given to demonstrate the difficulties of doing this and to show that the difficulties could be resolved by considering the purpose for which the object was to be distinguished as a stable entity despite possible vast physical changes occurring to it.

The analyst must design the method of forming and assigning names to the objects identified by the formal system. Uniqueness is the only structural requirement of a system of names though it may, with both advantages and disadvantages, embody the structural information.

Finally, the analyst must establish precisely how to link object to identifier and identifier to object, if the input to and output from his system are to relate meaningfully to the world. To some extent, this can be done with labels but rules of continuity and rules of synonymy are necessary too.

The LECOL language and system have been continuously in the background to the analysis. One of the purposes of LECOL is to make explicit the semantics upon which a formal system is based, so that they may be designed consciously, rather than by default, as at present, through want of a language in which to talk about the problem.

Acknowledgements

Many people must be thanked for their helpful comments on this paper. In particular, I will mention my colleagues at the London School of Economics, Frank Land, Sam Waters, Bob Davenport and Peter Mason, and my colleagues who, while I was working on the paper, were at IBM Peterlee, especially Terry Rogers, Peter Hitchcock, Pat Hall and Nigel Martin. Also I must thank the BCS Study Group on Databases for their helpful discussion of a draft version. Finally, the participants at the IFIP TC2 Working Conference in Nice 1977 led me to add ten pages to the opening of this paper in response to the discussions during the week, this being the last to be presented.

References

1. Stamper R.K. The LEGOL Project - A Survey IBM UKSC Report No.0041, Peterlee, 1976.

2. Kent, W. Describing Information (Not Data, Reality?) IBM Technical Report TR 03.012, Palo Alto 1976.

3. ANSI-SPARC/DBMS Study Group Interim Report American National Standards Institute, 1975

4. Stamper, R.K. 'Logical Structures of Files' in Data Organisation for Maintenance and Access The British Computer Society, 1970.

5. Stamper R.K. Information Batsford, London; Wiley, New York, 1973.

6. Caushay-Williams, R. The Double Criterion of Empirical Judgement, March 1961.

7. Quine, W.V.O. From a Logical Point-of-View Harper and Row, New York, 1953 (revised edition, 1961)

8. Dewey, J. How We Think Heath, London, New York, 1933

9. Popper, K.R. Logic of Scientific Discovery, Harper and Row, New York, 1935 (revised edition 1965)

10. Holland, J.W. Talking about Particulars Routledge and Kegan Paul Ltd, London, New York 1970.

11. Nijssen, G.M. A Gross Architecture for the Next Generation of Database Management Systems in (12).

12. Nijssen, G.M. (ed) Modelling in Database Management Systems North-Holland, Amsterdam and Oxford 1976.

13. Nijssen, G.M. (ed) Database Description North-Holland, Amsterdam and Oxford, 1975.

14. Nijssen, G.M. (ed) Concepts for the Conceptual Schema North-Holland, Amsterdam and Oxford, 1977.

15. Senko, M.E. DIAM as a Detailed Example of the ANSI SPARC Architecture in (12)

16. Todd, S.J.P. 'The Peterlee Relational Test Vehicle - a System Overview' IBM System Journal No4, 1976.

17. Mandil, S.M. The MP/3 Macro-Processor IBM UKSC0044, December 1973.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

THE ENTITY - RELATIONSHIPS MODEL - A BASIS FOR THE ENTERPRISE VIEW OF DATA

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

The entity-relationship model—A basis for the enterprise view of data

by PETER PIN-SHAN CHEN
Massachusetts Institute of Technology
Cambridge, Massachusetts

ABSTRACT

The concept of the enterprise view of data is very useful in the database design process and in the construction of conceptual schema. This paper discusses the use of the entity-relationship approach in describing and maintaining the enterprise view of data. Fundamental operations for changing the enterprise schema are presented. Finally, an example is given to show the differences between the entity-relationship approach and the data-structure approach in modeling the enterprise view of data.

INTRODUCTION

The subject of the logical view of data has attracted considerable attention in the past ten years. However, most researchers have focused on the user view of data. The need for studying the enterprise view of data was not recognized until recently. Different users of a database may have different views of the database, but the enterprise should have a unique and consistent view of the database. This is particularly important in designing a logically meaningful and consistent database. The concept of the enterprise view of data is very useful in the database design process and in the design of conceptual schema.

Enterprise view and database design

Database design is a process to organize data into a form which matches the underlying data model of the database management system. There are three major types of database management systems: network, hierarchical, and relational. In the network database management systems, which include Honeywell's IDS and UNIVAC' DMS-1100, data will be organized into different types of records and can be represented by a data-structure diagram[1] (see Figure 1). In the hierarchical database management systems, which include IBM's IMS, data will be organized into a form similar to but more restricted than the data-structure diagram. In the relational database management systems,[2] data will be organized into a set of tables (or "relations"). In general, to design a database is to decide how to organize data into specific forms (record types, tables) and how to access them. Up to now, there are very few tools available to aid the database design process. Usually, the database designer relies on his own intuition and experience. Thus, the resulting database may not satisfy company's objectives and may cause problems in company's operations.

Another related problem in database design is that the output of the database design process—the user schema (a description of the user view of data)—is not a "pure" representation of the real world. One of the reasons is that the database designer is restricted by the limited capabilities of the database management system. For example, the many-to-many relationships between entities are difficult to represent directly in some database systems. Another reason is that the user schema may contain some features related to the storage representation of the database. For instance, it may describe which record types can be directly accessed and how to access other record types. In addition, the user schema is usually designed to be efficient for a certain type of data processing operations. For example, the data about employees may be grouped into two record types, employee-master and employee-detail, to improve the retrieval performance. Therefore, the user schema is usually not a direct representation of the real world. This makes the user schema difficult to understand and difficult to change.

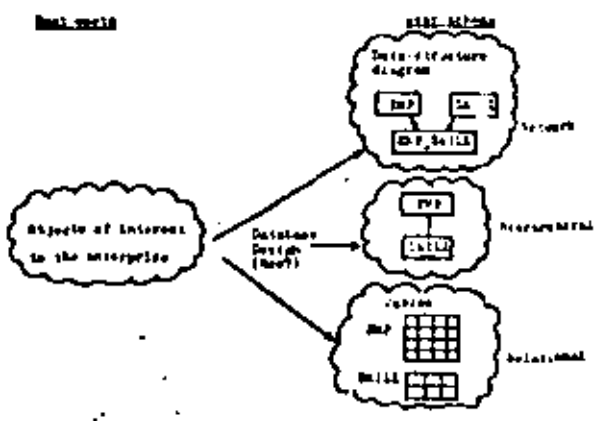


Figure 1—Conventional database design process

A possible solution to the above problems is to introduce an intermediate stage in the database design process: defining the enterprise schema, which is a "pure" representation of the real world and is independent of storage and efficiency considerations. The enterprise schema will then be translated into different types of schemata for different database management systems (see Figure 2). It can also be translated into several schemata for the same database management system to optimize different types of data processing operations. There are several advantages of this approach:

(1) The enterprise schema is easier to understand than a user schema since the former does not have the restrictions of the underlying database management system;
(2) The enterprise schema is more stable than the user schema, since some types of changes in the user schema may not require any change in the enterprise schema. If the enterprise schema needs to be changed to reflect the changes in the enterprise environment, the changes can be performed easily since efficiency and storage issues are not considered.

Enterprise view and conceptual schema

What is the difference between the enterprise schema and the conceptual schema proposed by the ANSI/X3/SPARC group?[3] Basically, they are very similar since both are descriptions of the enterprise view of data. In the SPARC's approach, the conceptual schema serves as the interface between the external schema (user view of data) and the storage schema (physical view of data) (see Figure 3). The requirement of serving as an interface between two other schemata may introduce some undesirable features into the conceptual schema. If this restriction on the conceptual schema is ignored, there is almost no difference between the conceptual schema and the enterprise schema. Therefore, the techniques discussed in this paper are also suitable for describing and maintaining the conceptual schema in the SPARC's architecture.

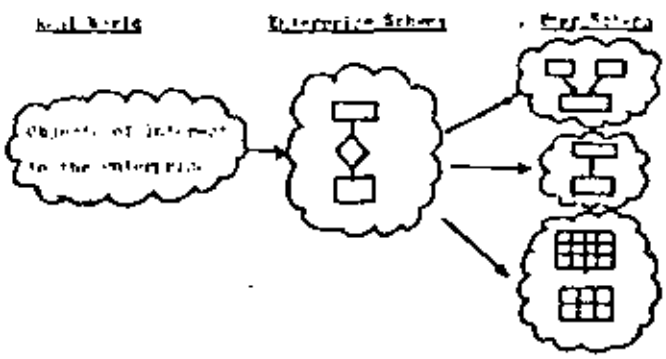


Figure 2—Enterprise schema as an intermediate step in database design

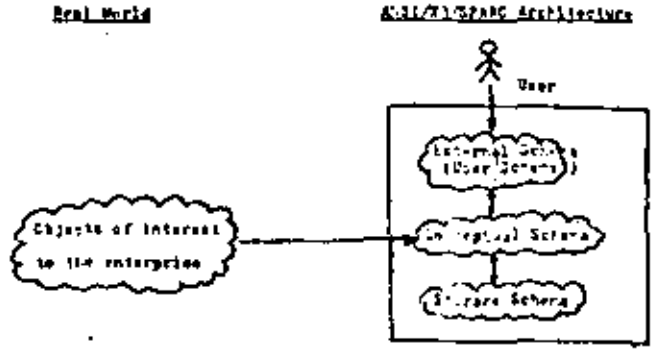


Figure 3—Enterprise view and conceptual schema

Approach used in the paper

In order to describe the enterprise view of data, a mental framework to model the real world is needed. Different people may be used to different mental frameworks. The mental framework used in this paper is the Entity-Relationship (E-R) model.[4,5] The E-R model and similar approaches[6-9] have been found useful in modeling the real world. A diagrammatic technique called the Entity-Relationship (E-R) diagram will be used in this paper to represent the enterprise view of data.

This paper is divided into three parts. The first part discusses how to use the E-R model and diagrammatic technique to describe the enterprise view of data. This is an extension of the work reported in Reference 5. The second part describes fundamental operations for changing the enterprise view of data. This is an area where very little work has been done. The operations proposed in this paper will be useful in maintaining the enterprise schema. The third part uses the E-R approach to analyze an example given by Bachman[10] concerning changes in the conceptual schema.

MODELING THE REAL WORLD USING THE ENTITY-RELATIONSHIP MODEL AND DIAGRAMMATIC TECHNIQUE

In this section, we shall use examples to show how to use the Entity-Relationship (E-R) model and diagrammatic technique to describe the enterprise view of data. A more formal definition of the model can be found in Reference 5.

It is assumed that the responsibility of defining and maintaining the enterprise schema belongs to a person called the *enterprise administrator*. The following is the suggested procedure for the enterprise administrator to define the enterprise schema:

(1) *identify entity sets of interest to the enterprise*

An *entity* is a "thing" which can be distinctly identified. According to the needs of the enterprise, entities can be classified into different *entity types* such as EMPLOYEE, STOCK_HOLDER. An *entity set* is a group of entities of the same type. In the E-R

Figure 4—Entity sets

diagram, an entity set is represented by a rectangular-shaped box (see Figure 4). The terms, "set" and "type," can be interchanged in the E-R diagram. The reader may use either one to interpret the E-R diagram.

There are many "things" in the real world. In addition, different enterprises may view the same thing differently. It is the responsibility of the enterprise administrator to select the entity types which are most suitable for his company.

(2) *identify the relationship sets of interest to the enterprise*

Entities are related to each other. Different *types* of relationships may exist between different types of entities. A *relationship set* is a set of relationships of the same type. For example, PROJ_EMP, which describes the assignment of employees to projects, is a relationship set defined on two entity sets, EMP and PROJ. A relationship set can also be defined on more than two entity sets. For example, PROJ_SUPP_PART is a relationship set defined on three entity sets PROJ, SUPP, and PART. In the entity-relationship diagram, a relationship set is represented by a diamond-shaped box with lines connecting to the related entity sets (see Figure 5). The "m" and "n" associated with the PROJ_EMP relationship in the E-R diagram indicate that the relationship is an m:n mapping. That is, each employee may be associated with several projects, and each project may have several employees. In certain companies, each employee belongs to at most one project, and the PROJ_EMP relationship is a 1:n mapping.

There are many types of relationships between entities. The responsibility of the enterprise administrator is to select the relationship sets (or types) which are of interest to the enterprise. He also has to specify the type of mappings (1:1, 1:n, m:1, or m:n) of the relationships.

(3) *identify relevant properties of entities and relationships (i.e., define value sets and attributes)*

Entities and relationships have properties, which can be expressed in terms of *Attribute-value* pairs. "Blue," and "4" are examples of *values*. Values can be classified into different *types* such as COLOR or QUANTITY. A *value set* is a group of values of the same type. An *attribute* is a mapping from an entity set (or a relationship set) to a value set (or a group of value sets). For example, "address" is an attribute which maps entities in the entity set EMP to values in the value set NAMES_OF_LOC. Note that we relax the constraint imposed in Reference 5 that the mapping from the entity set to the value set has to be a function (i.e., m:1 mapping). In other words, we now allow that an attribute (such as address) can have several values (such as locations) for the same entity (employee). This relaxation in the definition of attribute will make the changes in the enterprise view simpler. This point will become clear in the next section.

In the E-R diagram, a value set is represented by a circle, and an attribute is represented by an arrow directed from the entity set (or the relationship set) to the desired value set(s) (see Figure 6). After selecting entity sets and relationship sets, the enterprise administrator identifies the attributes and value sets which are relevant to the company's operations.

Figure 5—Relationship set

The three steps stated above cover a major part of the enterprise schema. For simplicity, we shall not discuss in this paper other issues related to the enterprise schema such as integrity constraints.

To design a database, the enterprise administrator first draws an E-R diagram such as the one shown in Figure 7. He then drew the attributes and value sets for each entity set and relationship set. The E-R diagram is then translated into a data-structure diagram or a set of tables ("relations") (see Figure 2). The rules and procedures used in the translation process were discussed in Reference 5. Here, we shall investigate how to change the enterprise schema (the E-R diagram) itself.

MODIFICATION OF THE ENTERPRISE VIEW

Although the enterprise schema is more stable than a user [illegible], it still needs to be changed from time to time

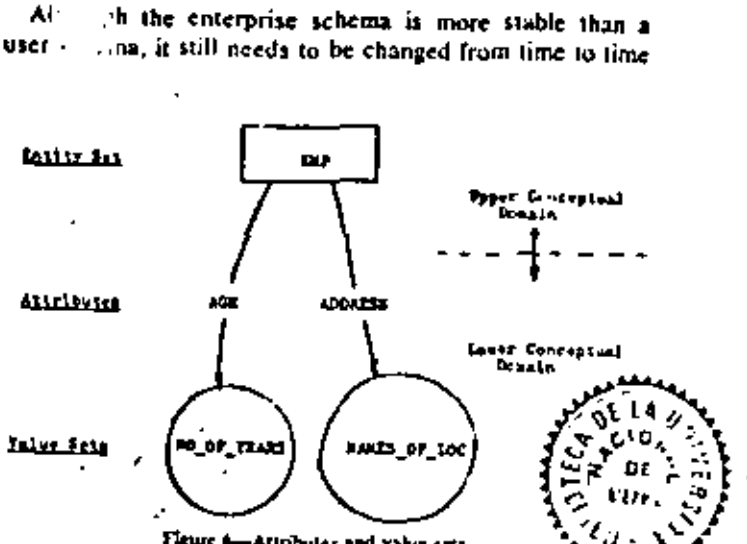


Figure 6—Attributes and value sets

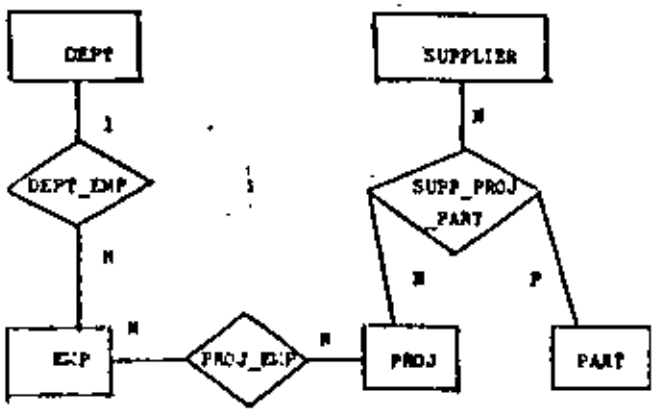


Figure 7—An entity-relationship diagram (with entity sets and relationship sets only)

to reflect the changes in the enterprise environment. Excepting a paper by Bachman,[9] very little work has been done in this area. In this paper, we use the E-R model as a basis for analyzing different types of changes in the enterprise view of data. We not only propose a set of operations but also analyze the consequences of these operations.

There are five basic types of operations: *add*, *delete*, *split*, *merge*, and *shift*. The first four operations are applicable to entity sets, relationship sets, attributes, and value sets. The *shift* operation is used when the enterprise administrator would like to view a value set in the old enterprise schema as an entity set in the new schema or vice versa. It is useful to think that the E-R diagram consists of two conceptual domains: (1) the upper conceptual domain which consists of entity sets and relationship sets; (2) the lower conceptual domain which consists of attributes and value sets. We shall discuss the first four operations in both the upper and lower conceptual domains. Finally, we shall discuss the shifting an entity set from the upper conceptual domain to the lower conceptual domain and the shifting a value set in the opposite direction.

Operations in the upper conceptual domain

The following are the basic operations applicable to entity sets and relationship sets:

(1) *Split an entity into several subsets*
For instance, the entity set EMP in Figure 8a can be split into two entity sets: MALE_EMP AND FEMALE_EMP in Figure 8b. The consequence of this operation is that the relationship sets associated with the entity set may also have to be split. For example, PROL_EMP is split into PROL_M_EMP and PROL_F_EMP (see Figure 8b).

(2) *Merge several entity sets into one entity set*
This is the opposite operation of (1). The consequence is that the related relationship sets may have to be merged.

(3) *Split a relationship set into several subsets*
An example of this operation is: the relationship set PROL_EMP in Figure 8a can be split into two relationship sets, PROL_MANAGER and PROL_WORKER, in Figure 8c. Note that the type of mapping in the new relationships may be different from that in the original relationship. For instance, the mapping in PROL_MANAGER is 1:n while the mapping in PROL_EMP is m:n.

(4) *Merge several relationship sets into one set*
This is the opposite operation of (3). Note that these relationship sets have to be defined on the same group of entity sets.

(5) *Add a new entity set*
For example, a new entity set called SUPPLIER may be added to the E-R diagram in Figure 8a. The result is shown in Figure 9a. Note that it is possible to have stand-alone entity sets in the enterprise schema, although in many cases relationships between the new entity set and the existing entity sets are established immediately (see the next operation).

(6) *Add a new relationship set*
We may add a new relationship set for the new entity set such as the relationship set PROL_SUPP in Figure 9b. We may also add a new relationship set for existing entity sets such as the relationship set PROL_MANAGER in Figure 9b.

(7) *Delete an entity set*
For instance, after deleting the entity set EMP in Figure 9b, we have Figure 9c. The consequences are: (i) the relationship sets related to the entity set are

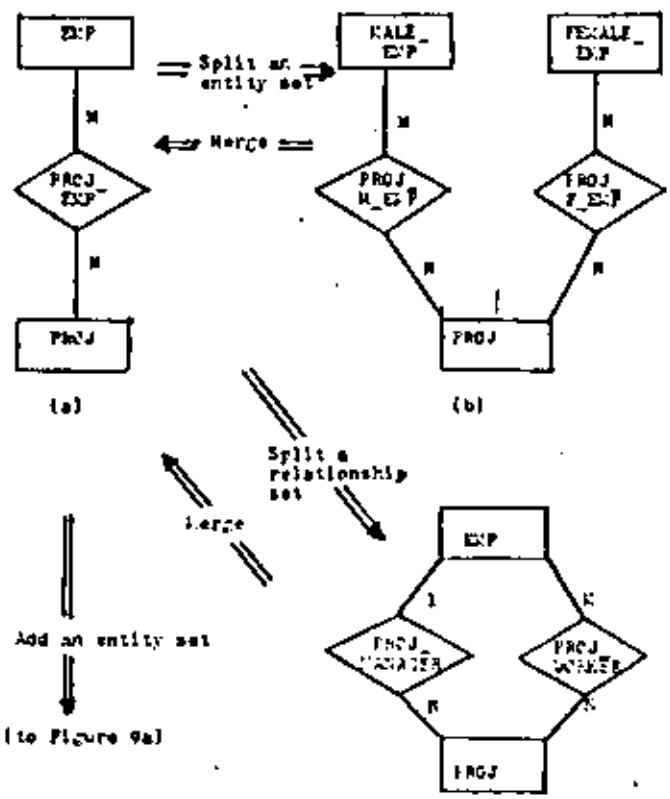


Figure 8—{Split / Merge} {Entity / Relationships} Sets

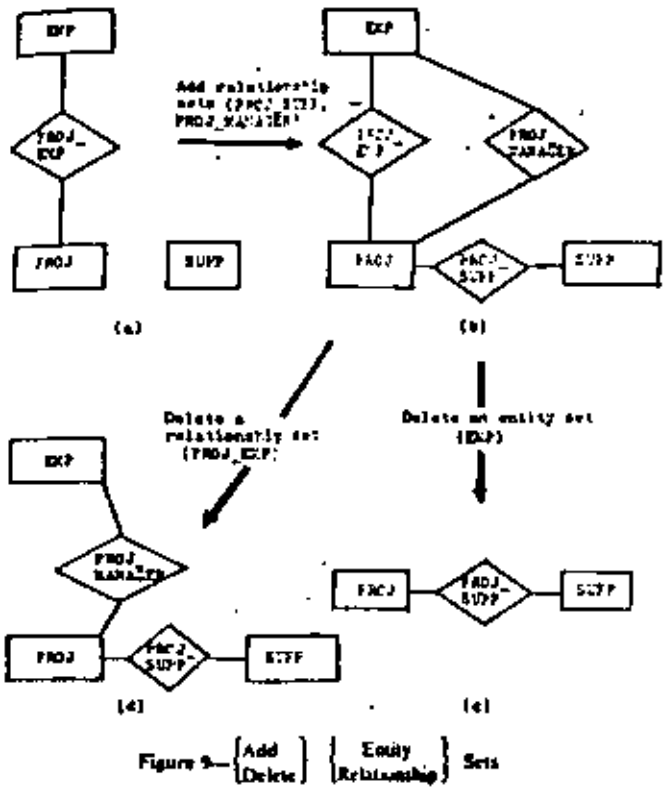


Figure 9—{Add / Delete} {Entity / Relationship} Sets

also deleted; (ii) attributes related to the deleted entity set and related relationship sets are also deleted.

(8) *Delete a relationship set*
An example is: delete the relationship set PROJ_EMP in Figure 9b, and the result is shown in Figure 9d. The consequence of this operation is that the attributes of the relationships are deleted (not shown in Figure 9d).

Operations in the lower conceptual domain

Assume that the entities in the entity set EMP have two attributes, LEGAL_NAME and PHONE, which map the entities to the value sets NAME and PHONE_# (see Figure 10a). We shall use these attributes and value sets as the basis for the discussion of the following operations:

(1) *Add a value set*
For example, a new value set called DOLLARS may be added to Figure 10a. The result is shown in Figure 10b. Usually, this operation is followed by an "add attribute" operation.

(2) *Delete a value set*
After deleting the value set PHONE_# in Figure 10a, we get Figure 10c. The consequence is that all attributes associated with this value set will be deleted.

(3) *Split a value set into several subsets*
The value set NAMES in Figure 10a may be [illegible] LAST_NAMES in Figure 10d. The consequence is that attributes related to the value set may have to be adjusted. Although the attribute LEGAL_NAME is not split in Figure 10d, it is possible to split it into two attributes: LEGAL_FIRST_NAME and LEGAL_LAST_NAME. It is the responsibility of the enterprise administrator to make this decision.

(4) *Merge several value sets into a value set*
This is the opposite operation of (3).

(5) *Add an attribute*
For instance, Figure 11b is obtained by adding the attribute OTHER_NAME to Figure 11a.

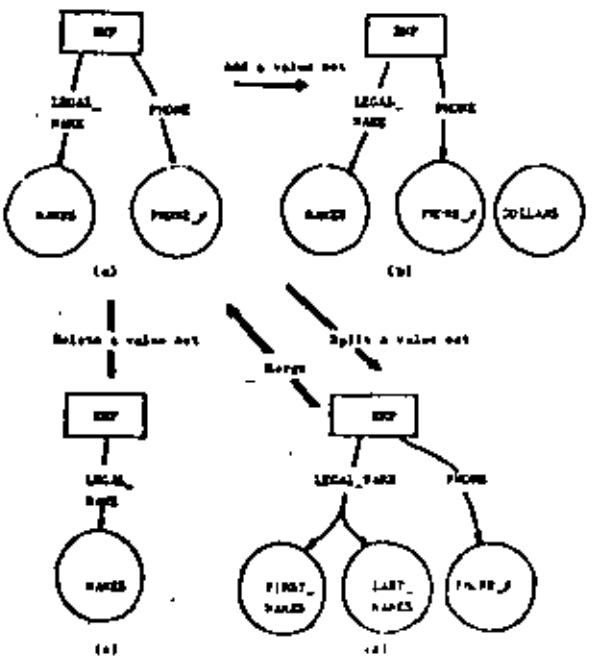
(6) *Delete an attribute*
Deleting the attribute LEGAL_NAME from Figure 11a, we have Figure 11c. The value set associated with the attribute will be deleted by another operation ("delete value set") if desired. In some cases, the value set may be still associated with other attributes (see Figure 11c).

(7) *Split an attribute into several attributes*
For example, Figure 11d is obtained by splitting the attribute PHONE in Figure 11a into two attributes, OFFICE_PHONE and HOME_PHONE.

(8) *Merge several attributes into one attribute*
This is the opposite operation of (7). The attributes have to be defined on the same entity set (or relationship set).

Operations between two conceptual domains

Assume that there are two entity sets (EMP and PROJ), one relationship set (PROJ_EMP), four value sets (NAMES_OF_PLACES, SOC_SEC_#, PHONE_#, and



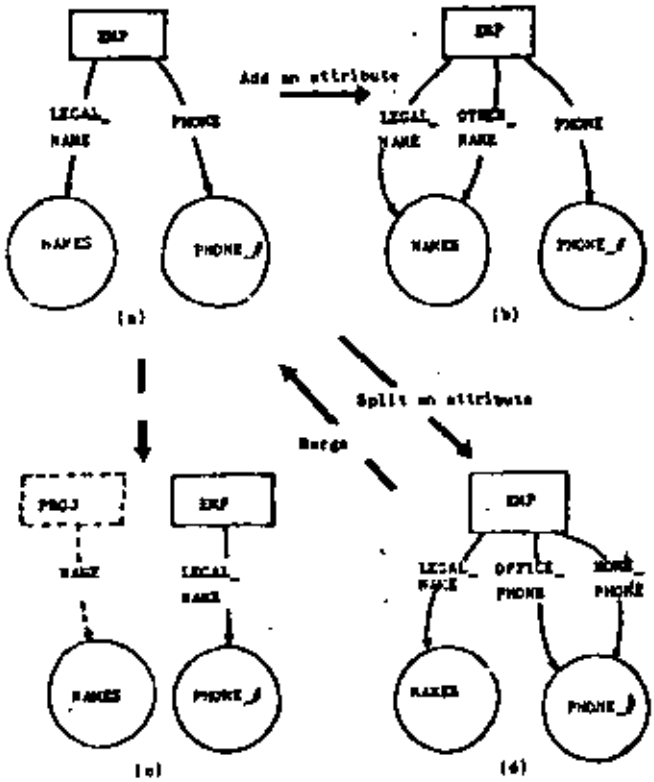


Figure 11—Operations on attributes

PROJ_NAMES), and four attributes (ADDRESS, SOC_SEC_NO, PHONE, and NAME) as shown in Figure 12a. We shall use them as the basis for the discussion on the following operations:

(1) *Shift a value set from the lower conceptual domain to the upper conceptual domain*
When the enterprise environment changes, it may become natural to view PLACE as an entity set instead of a value set. Thus, in Figure 12b "ADDRESS" becomes a relationship set, and "PLACE" has an attribute "NAME" which points to the value set NAMES_OF_PLACES. Since PLACE is an entity set, we may establish new relationships of it with other entity sets such as PROJ or add more attributes and value sets to describe properties of "places."

(2) *Shift an entity set from the upper conceptual domain to the lower conceptual domain*
When the enterprise environment changes again, it may become natural to view PROJ as a value set instead of an entity set. In Figure 12c, PROJ is deleted from the upper conceptual domain, and the relationship set PROJ_EMP becomes the attribute INVOLVED_PROJ. The entity set PROJ in Figure 12b may have been associated with several value sets, but only the value set PROJ_NAMES which is used to identify the entities PROJ remains in the lower conceptual domain.

ANALYSIS OF AN EXAMPLE

In a recent paper, Bachman[10] uses data-structure diagrams to illustrate the changes in a conceptual schema. In this section, we shall first state his example and then use E-R diagrams to interpret his example.

Description of the example using data-structure diagrams

The following is a simplified version of Bachman's example:

(a) In the beginning, the enterprise administrator declared a conceptual schema as shown in Figure 13a. The reader is assumed to have some knowledge of the data-structure diagram.[1] Simply speaking, a rectangular-shaped box represents a record type, and an arrow represents a data-structure-set (i.e., 1:n relationship between record types). In Figure 13a, there are two types of conceptual records, COMPANY and PERSON, and a data-structure-set "a" representing the fact that each person is associated with exactly one company and that each company has a set of personnel.

(b) Later, the enterprise administrator recognized that the personnel of the company were persons in their own right. This fact may be discovered at the merger

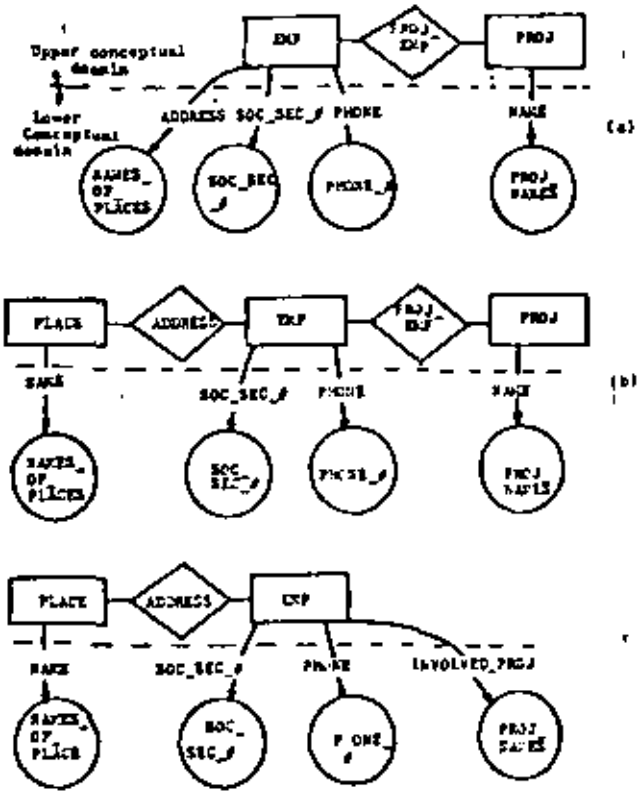


Figure 12—Shifting a set from the upper conceptual domain to the lower conceptual domain and vice versa

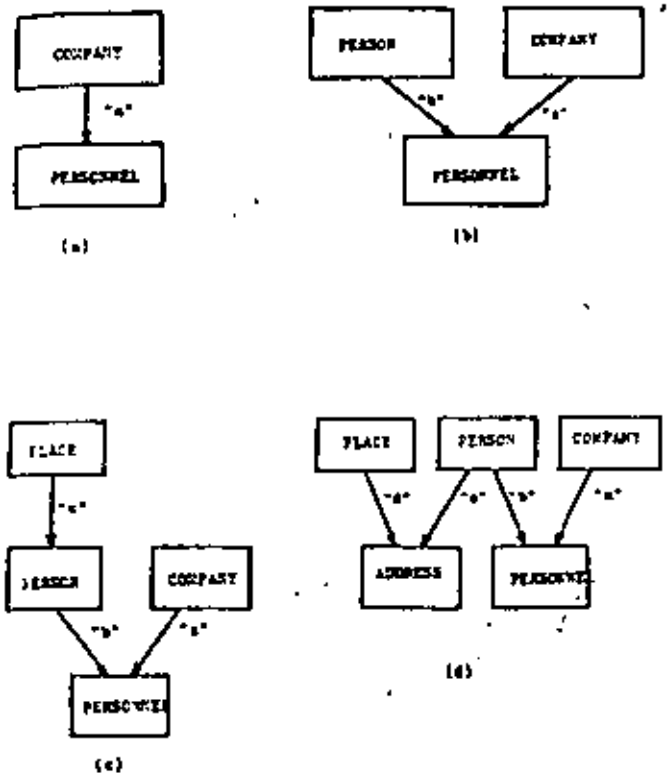


Figure 13—Expressing changes in the enterprise view using data-structure diagrams

of several companies that some of the persons held two jobs and were personnel to two of the merged companies. Figure 13b illustrates the data-structure diagram for the new conceptual schema. Basically, the old personnel type record has been split into two record types, PERSONNEL and PERSON. The "PERSON" has attributes NAME and ADDRESS (not shown in the figure).

(c) After a while, the enterprise administrator decided to factor the address of residence out of the person record. Figure 13c illustrates the addition of the "PLACE" conceptual record type and the data-structure-set type "c." It was also assumed that each person has a unique address (place).

(d) It is now recognized that people move from place to place and that it is desirable to know current address as well as past addresses. Another reason may be: it is discovered that a person may have more than one address. In either case, a new conceptual record type ADDRESS is added to the conceptual schema (see Figure 13d).

Analysis using entity-relationship diagrams

In the following, we shall use E-R diagrams to explain the above example:

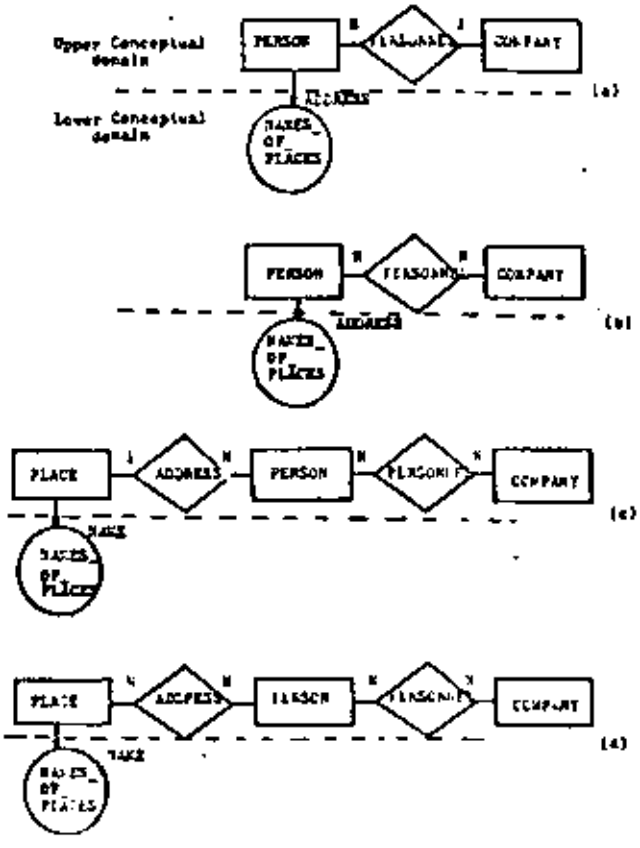
(a) The E-R diagram in Figure 14a is corresponding to

two types of entities, PERSON and COMPANY, in the enterprise view. Since the mapping between COMPANY and PERSON is 1:n, the relationship set PERSONNEL is represented by a data-structure-set "a" in Figure 13a.

(b) Figure 14b is the corresponding E-R diagram for Figure 13b. Since the relationship set PERSONNEL is an m:n mapping, it is represented by a relationship record type PERSONNEL and two data-structure-sets "a" and "b" in Figure 13b. Note that Figures 14a and 14b have the same entity sets and relationship set in the upper conceptual domain, and the difference is the type of mapping between the entity sets.

(c) Now the enterprise administrator prefers to view "PLACE" as an entity set rather than a value set. Thus, we have Figure 14c. The attribute ADDRESS in Figure 14b becomes a relationship set in Figure 14c. Since the mapping between PLACE and PERSON is 1:n, the relationship set ADDRESS is represented by the data-structure-set "c" in Figure 13c.

(d) The enterprise administrator discovers that the mapping between PLACE and PERSON is an m:n mapping instead of a 1:n mapping. The new enterprise view is represented by Figure 14d. Since the mapping is m:n, the relationship set ADDRESS is represented by the record type ADDRESS and two data-structure-sets "d" and "e." Note that Figures 14c and 14d



are almost the same except that the type of mapping between PLACE and PERSON is different.

In general, the E-R diagram is easier to use to analyze the changes in the enterprise view than the data-structure diagram. Bachman also raised the issue of the ambiguity in Figure 13d: If one wants to modify a person's address, does he have to create a new "address" record or to change the name of the place where the person is living? This question can be easily answered using the E-R approach. Consider Figure 14d. Since the PLACE is an entity set, to change a person's address is to change the relationship between the person and "his place." We should not change the name of the place where the person is living since "NAME" and "NAMES_OF_PLACES" are used to describe a property of the PLACE entities (see Figure 14d).

SUMMARY

The enterprise schema is useful as an intermediate step in database design. In this paper, we have shown how to use the entity-relationship model and diagrammatic technique to describe the enterprise schema. Since the enterprise environment changes from time to time, the enterprise schema will have to change to reflect these changes. Five basic types of operations (add, delete, split, merge, and shift) which are useful in modifying the enterprise schema have been presented, and the consequences of these operations have been discussed. Finally, we have used an example to analyze the differences between the entity-relationship approach and the network approach in modeling the enterprise view of data.

REFERENCES

1. Bachman, C. W., "Data Structure Diagrams," *Data Base 1*, 2, Summer 1969, pp. 4–10.
2. Codd, E. F., "A Relational Model of Data for Large Shared Data Banks," *Comm. ACM 13*, 6, June 1970, pp. 377–387.
3. ANSI, *Interim Report of ANSI/X3/SPARC Group on Database Management Systems*, ANSI, February 1975.
4. Chen, P. P., "The Entity-Relationship Model," (abstract), *Proc. 1st Very Large Database Conf.*, Framingham, Mass., Sept. 1975, ACM.
5. Chen, P. P., "The Entity-Relationship Model: Toward a Unified View of Data," *ACM Trans. on Database Systems 1*, 1, March 1976, pp. 9–36.
6. Moulin, P., J. Randon, M. Teboul, et al., "Conceptual Model as a Database Design Tool," *Proc. IFIP TC-2 Working Conf.*, Jan. 1976, Black Forest, Germany, pp. 459–479.
7. Hall, P., Todd M. Owlett, "Relations and Entities," *Proc. IFIP TC-2 Working Conf.*, Jan. 1976, Black Forest, Germany, pp. 430–458.
8. Deheneffe C. and H. Hennebert, "NUL: a Navigational User's Language for a Network Structured Data Base," *Proc. ACM 1976 SIGMOD Conf.*, Washington, D.C., June 1976, pp. 133–142.
9. Tozer, E. E., "Database Systems Analysis and Design," Technical report, Software Sciences Limited, England, April 1976.
10. Bachman, C. W., "Trends in Database Management—1975," *Proc. AFIPS 1975 NCC*, Vol. 44, AFIPS Press, Montvale, N. J., pp. 569–576.

DISEÑO DE ESTRUCTURAS DE DATOS

ARTICULO

INFOLOGICAL MODELS AND INFORMATION USER VIEWS

EXPOSITOR:

ING. DANIEL RIOS ZERTUCHE

MAYO, 1984.

Inform. Systems Vol. 5, pp. 17–32
© Pergamon Press Ltd., 1980. Printed in Great Britain

22

0306-4379/80/01 0017-16 $02.00/0

INFOLOGICAL MODELS AND INFORMATION USER VIEWS

BÖRJE LANGEFORS
University of Stockholm, Department of Administrative, Information Processing,
Fack, S-1045 Stockholm, Sweden

(*Received* 15 *January* 1979; *in revised form* 1 *June* 1979)

Abstract—The study of data systems as information systems put into focus the role of data as representations of information in the sense of knowledge about some slice of the world. This information system-view—or infological view—made it clear that the data alone cannot "carry" information. They can only, at best, give rise to information in the minds of people and only in those people who hold a suitable frame-of-reference or world view, or "receiving structure", in their mind. Thus the infological perspective had to be widened successively from a concern merely with information representation, structuring, and exploitation to the study of social, socio-psychological and socio-linguistical aspects and of "object system", job design and other socio-technical issues.

The term "user view" is employed widely in recent data base work. The use of the term "user view" suggests such an "infological" perspective. However, a closer look at how the term is actually used indicates a much more delimited interpretation which focuses on representational aspects and processing. This is also made explicit, to some degree, by the usual formulation "user view of the data" rather than, for instance, "user view of the world".

In this paper a brief study is made of the two aspects of user views,

(i) the *infological/conceptual aspect*, which is concerned with how conception relates to data and information, and to reality, and

(ii) the "*datalogical*" *aspect*, which is concerned with the selection of data from a data base and the rearrangement of them to suit a "user view" of the data (as seen from the application programmer).

The infological aspect is illustrated through a discussion of some of my own earlier results which are here brought together. The datalogical aspect is exemplified by some quotations from the most recent data base literature, as well as by some earlier results of my own.

The term "user view" is frequently used in the datalogical sense whereas the infological or information-system-theoretical studies often have addressed questions that have to do with the infological/conceptual aspects of "user views", without employing that term. In this paper some aspects of the problem of infological/conceptual user views are treated with a view to gain understanding of how both aspects of user views affect the design and use of information systems and data bases. Illustrations are taken from the author's own earlier work, for three reasons: (i) they were most easily available, (ii) they are directly associated with the problem at hand, and (iii) they have earlier been scattered over several works and it was useful, for the purpose of the present discussion, to bring them together.

INFORMATION SYSTEMS THEORY (IS THEORY) VS TRADITIONAL DATA PROCESSING CONCEPTS

In traditional data processing an information item is usually assumed to be a number which is to appear at a certain part of an application program. However, when taking on to study the problem of how to design data processing from another perspective, as a function within an information system IS (Langefors[1–3]) it was found that it is necessary to look closer into how data can provide information to people and to organizations. It becomes immediately obvious that if data are what are processed by computers (or other means) then the information which people get from the data is something distinct from the data. If data may aid people in making decisions or performing actions, this must mean that the data *inform* people in the sense of making something known to them. It is clear then that the data must represent something that can also be expressed by natural language. Clearly then, in this sense, written sentences of natural language (as well as spoken or recorded sentences of any kind) are also data, that is, they are sets of signs representing knowledge or information.

It is interesting to note the perspective advanced above is similar to ideas treated by philosophers at least as far back as Aristotle, and also, more recently, by, e.g. Russel, Wittgenstein, Carnap and Bar-Hillel. It took some time, however, before these ideas were recognized as useful to the understanding of data processing systems. Also, recent development in cognitive psychology and psycholinguistics is providing new insights into this area (Bar-Hille[4], Carnap[5], Piaget[6], Wittgenstein[7], Bower and Anderson[8], Lyons[9]).

The perspective of information systems theory and infology studies has been that to design the right data and data systems one has to begin with the questions about the information or knowledge that the data are to represent. This means that the information content of the data in the system has to be defined and described in a way that is independent of how the data *are* handled in the system. The information is described in terms of the views of the world that the information users hold. Information modelling, thus, is world modelling. One consequence of the data independent data description is that the data system (files, data base, programs) may be restructured without the need for changes of the information specification.

Recent data base work and structured-programming work has stressed the importance of describing the data

independently from the representational details. For instance in Codd[10], it is stated:

> "The relational view (or model) of data..... provides a means of describing data with its natural structure only—that is, without superimposing any additional structure for machine representation purposes".

While this piece of text does not speak of the information conveyed by the data, the reference to "its natural structure" suggests a somewhat information-oriented perspective. However, the starting point here is the data and its handling by systems whereas the infological perspective is that of people and their world. Indeed, the information specified may not necessarily come to be handled by computers at all. It seems that the focus on data—rather than on information—has some consequences. Thus some of the basic concepts of the relational data base theory, such as normalization and join operations, are in fact based on arguments that are concerned with how the data are viewed as being stored and processed by the system, in spite of the expressed intention to consider the "natural structure only". Likewise, recent data base work on "user views" turn out to be mainly concerned with views on how the data are arranged for processing, as we shall see. As to the perspectives of structured programming we may quote from Jackson[23]:

> "The problem environment is the real world...The customer file is a model, intelligible to the computer, of the set of customers;...."

This text also suggests an informational (or infological) perspective but, on the other hand, the phrase "intelligible to the computer" indicates that the thoughts quickly touch upon the processing of the data representing the information. As we shall see—also in this work—the view of the data and their structure is similar to the "user view of the data" as discussed in the data base works.

The intention behind this paper is to propose the widening of the conceptions of "user views of data" and the "natural structure of the data" that may be obtained by merging these conceptions with infological conceptions of information structure and the pre-knowledge of the various intended users.

We shall see that "user views" are not merely of importance to the way data are to be organized in the store. They are crucial to the possibility for the users to interprete the data. Thus, not only must the data be arranged and ordered to allow efficient processing of the data, according to some view. The data themselves will need to be changed in order to represent the same information to users with a different view. Moreover, some people may have such a view that they are quite unable to hold any information whatever data one might try to use. This, of course, is critical; it is a threat to the very assumption of large IS or large, shared data bases.

SOME BASIC INFOLOGICAL CONCEPTS AND TERMS

The basic concepts or models of information systems theory and infology are intimately associated with semantical questions and user views. Thus, before discussing some aspects of information user views we review some of these basic concepts.

Particular content of a single data term

"Let's turn back to the question of how an elementary data item, such as a number, can represent information. It is clear that the traditional view, that the treatment of the data item by a certain program determines the information, cannot provide the full answer. If certain data (for instance a certain printed sentence) are to represent some knowledge of the world, this must be achieved through reference to phenomena in the world and not merely through its relation to any specific computer program. Thus, given a data item or term, for instance the number 17, how can it inform about the world?

Two kinds of informational questions arise in connection with any individual data term:

(A) What other data terms are needed in order to combine with the particular term so that together they may provide information?

(B) How may one retrieve any specific term as stored in a data system? (Retrieval may be requested by various application programs or by various users at computer terminals.) (Langefors[1–3]).

In order to discuss the two questions presented above we will introduce some terminology. We shall say that a group of data which is necessary and sufficient for the retrieval of a specific term "t" is a "*retrieval determinant*" (or key) for "t" and a group of data which is necessary and sufficient in order to supplement "t" to provide information is an "*information determinant*" or *i-determinant* for "t".

It is seen that some kind of dependency must exist between a data term and its retrieval determinant and, likewise, a dependency must exist between the term and its i-determinant.

Furthermore, we may conclude that the sources of these dependencies between the data are dependencies between the phenomena that the data refer to. These dependencies give rise to corresponding mathematical dependencies among the data (as such dependencies are studied in recent data base work).

Let us consider a specific data term, such as the number 17 or, more precisely, the printed sign "17". What does it tell us? It is natural to say that, as standing alone, it does not convey any information. On the other hand it does have meaning in the sense of referring to a concept, the concept of a number. Thus, we seem to require something more than meaning (in some sense) from a piece of information. Now, let us assume that we are told that the term "17" is a quantity on hand in a store. This attaches some more specific meaning to the term. Accordingly if the data term "17" is supplemented with the data term "Quantity-on-Hand, 17" some more precise specification of meaning is conveyed. But still we do not obtain information. Instead, if we are presented with the sentence "The quantity in store of articles of type A is 17 pieces" we feel having been informed. We might draw some conclusions from such a statement, provided we perceive what it will look like in the stock room or how we might proceed if we would want to

verify the sentence. Thus it adds to our *view of the world*. It seems then that a text string, thus a group of data terms, in order to convey information not only must have meaning, it must have a truth value also—it must state a fact and would be regarded as false if contradicted by real facts. More generally, it should present knowledge of a system state.

It appears now that the question whether some data convey information is either related to formal logic or asks about a view of the world. Furthermore, in logic there is sometimes made a distinction between a *declarative sentence* (which is intended to state a fact) and the *proposition* which is formulated or represented by the sentence. The proposition is "the cognitive content" of the sentence and "the thought of a fact" which may or may not prevail. It is clear, now, that the proposition (in the logical sense of above) is the *information to be conveyed* by the sentence and the sentence is a group of data used to represent that information (the proposition). However, even if a proposition is a paradigm case of an information unit, as we found, it may not be the only kind. Conceiveably, data might be used to represent information that is not of the clear-cut logical kind of a proposition. A view of the world may be something else than a proposition. This we do not pursue here, we only want to make the declaration that until further study has been done on this issue, we will use the concept of logical propositions merely as the simplest case of information units—but not necessarily the only possible type (Langefors[1, 3]). For this reason the term "message" (more recently i-message) for a piece of information or knowledge was chosen. The term "proposition" was seen as a special case of an "i-message". Clearly, the question of what is a piece of information would have been a lot easier if one simplified oneself to just a "proposition". Then, simply, it would have been possible to refer to textbooks in logic. But important practical IS design questions fall outside such a narrow framework so we have to be more general, for practical reasons. Perhaps it would be better to use the term "proposition" instead of "message" but then with a somewhat generalized meaning. Better still, perhaps, we could use "i-proposition" to indicate this generalization.

Elementary information units-e-messages

A message, like a proposition, may be a composition of other messages. And a view may be a composition of views. A message (or a proposition) which cannot be decomposed into several messages is referred to as an elementary message (*e-message*) (or an elementary proposition—or "e-proposition"—respectively). Correspondingly a data group (such as a sentence) may represent a compound message in which case it may be decomposed into *elementary sentences* (e-sentences) or *e-records*. An e-record or an e-sentence is data representing an e-message (according to some syntax/semantics). Obviously an e-message is "*irreducible*" in the sense described above.

We are now in a position where we could say that to find the i-determinant, that is, to determine the additional data terms needed as the supplement of any given term (such as "17") in order to represent a piece of information, we have to identify (at least) an elementary sentence of natural language, containing the given term ("17"). However, this would require us to have the computer handling, and interpreting, natural language texts. This was clearly impractical in 1966 and it still is. For instance, natural language is highly ambiguous. Consequently, to study data one needs to have some formal data models and information models for e-messages and e-records (Langefors[3]). Because of the close relationship between e-messages and elementary propositions as well as between e-records and elementary sentences, one could turn to formal logic for help. This would need some caution, though, because it was found necessary not to be restricted, already by definition, to logical e-messages (propositions) only. Also, because it is necessary to handle information about information we cannot accept to be locked in within the bounds of first order formal predicate calculus and because the time dimension is also significant to an IS this is another reason for using a wider frame than traditional, formal logic. In spite of this, we may often find that formal logic can usefully be employed in many special cases.

Elementary sentences or e-records

Before proceeding let's look at some alternative sentences formulating the same e-message as the sentence proposed on p. 18, 2nd column. This should make it clear that some formal modelling is needed.

(1) "There remain 17 pieces of article type A for disposal"
(2) "Of article A there are 17 pieces on hand"
(3) "We are now left with 17 pieces"

We notice that the sentence (3) clearly presupposes some *particular* context (in addition to the "general context" or "frame-of-reference" that is always necessary).

From formal logic one may be led to the formal sentence, or formula,

(4) "Qty-on-Hand (A, 17)"

In (4) Qty-on-Hand is assumed to be the predicate "quantity-on-hand of a specified article type". Such formulations as (4) do not appear convenient enough for the general information system user. In addition the use of free variables and quantifiers of predicate logic appear unnecessarily complex for many data design situations (though they are sometimes hard to avoid). Also, as stressed above, it was not felt to be acceptable to be totally constrained to the use of predicate logic. Hence formal logic was rather seen just as a source of suggestions for a general information model. It did not appear to be feasible without some modification and amendment Langefors[3]. It is worth noticing, furthermore, that the use of the common form of predicate logic statement, as in (4), calls for a particular syntax/semantics description for each individual predicate

The structure of e-messages

Based on arguments like those above it was concluded (Langefors [1, 3]) that an e-message, however otherwise formulated must *make known*, one way or the other:

(a) What object or entity it is intended to inform about (the "Subject" of the e-message)

(b) What it makes known about that entity, the characteristicum part (or property part) of the e-message

(c) A time during which the Subject holds the characteristic

(d) Because the above states the minimum content of a message it is important also to assign a name (identifier) to each e-message (so that other messages may be provided which add information about this message).

It is important to recognize that *how* the aspects (a), (b), (c), (d) are made known is left totally open. Thus any e-message may be structured in many ways as long as the three basic references (*Subject Reference, Characteristicum Reference,* and *Time Reference*) are established. A group of data that may be used to make known the e-message is an *e-record* or an *e-sentence*. Obviously many different e-sentences may be used to represent an e-message. *Remark*: The individual data terms or items, that make up an e-record may be put in distinct places in the "physical data structure". When we want to remind on this fact the term "e-entry" may be used, rather than "e-record" (borrowing "entry" form CODASYL).

If one looks at one of the example sentences above: "Quantity-on-hand of article type A is 17 pieces" it may be concluded that the object it informs about is "article type A" and the property or characteristic that it states about that object is "quantity-on-hand = 17 pieces". Furthermore, the time stated (implicitly) is "now". Thus we may specify, more formally:

(1) e-message identifier: em 1
Subject: Article-Type, A;
Characteristic: Quantity-on-Hand, 17 pieces;
Time: Present;

This structure may be presented in many other ways provided, of course, that the form used is made known through some sort of format statement. Some such forms may be as shown below, Nos (2–4).

(2) Format: Identifier: Subject Reference; Characteristicum Reference; Time Reference;;
em 1:
Article-Type, A; Quantity-on-Hand, 17; Time, Present;;

or

(3) Format: Identifier: Characteristicum-Type (Subject Reference, Time Reference, Value);
em 1:
Quantity-on-Hand (Article-Type, A; Time, Present; 17 pieces).

(4) Format: Identifier: Characteristicum-Type (Object Class, Time-Type)
((Obj-id); (Time-Value); (Quantity));
em 1:
Quantity-on-Hand (Article-Type, Time, Value) (A, Present, 17 pieces);

It may be noted that each of the message-representations (or sentences) presented above are in a sense both complete and minimal). They are minimal (or irreducible) in the sense that if any of the terms is deleted (hence ignoring the associated reference or knowledge) then that which remains does not provide information. The reader should verify this for himself. In our particular illustrative case this means that no logical proposition is represented when any single term is deleted. The messages are complete, each of them, in the sense that they make known something that may be true or false and that thus is a proposition. For instance if the fact is that at present there are 18 pieces in inventory, of article type A, then the proposition is false. We may thus say that any of the message representations above (1, 2, 3 or 4) defines a complete minimum supplement to the single term "17" and, thus, makes known the information which "17" takes part in. In other words, any of the e-message representations above defines an i-determinant of the term "17".

It follows from the discussion that any of the above e-message representations (as printed) can be regarded as an e-record for the e-message em 1. Any other group of data which can be formally mapped into any e-message representation for em 1 may be used as an e-entry for representation of em 1 in a data system (provided the mapping procedure is defined).

The e-message representation No. (4), above, has a form which has separated the variable terms as the "value tuple" (or "tuple" for short): (A, Present, 17 pieces). It is seen that any e-message (and e-record) contains a relation instance as a proper part. Thus any e-message type defines a (data base) relation (once value domains have been chosen).

Illustration of some basic concepts

Infological studies call for a large number of concepts, as a result of analyzing the concepts of information into the aspects: representations, conception, and reality and into the type and instance levels as well as into distinct levels of aggregation and definition. All the resulting concepts have been found necessary. They are treated explicitly, almost one by one, in the systematic information analysis. They call for a corresponding, large number of terms. Of course, this requires some mental effort from the analyst before he can make use of the concepts, as the reader will have noticed. This is a reflection of the complexity of the problem of info-modelling. To keep down the number of entirely distinct terms a systematic set of prefixes has been introduced. It may be useful to illustrate the employment of the terms by means of an example, as an aid to the user's comprehension of the infological frame of reference employed here.

The natural-language sentence

M1: "The customer C ordered 5 pieces of article A today" may be interpreted as representing a real-life event or fact. If that is so the sentence M1 may be said to convey a piece of information which is the knowledge of this fact. The fact, in this case, is an event in the real world. The piece of information is referred to as a *message*. (In this specific case it is a *proposition* in the sense of logical texts.) A structure of data that may be used to *represent* a message is a *record*. The natural-language sentence M1 is thus a record, by definition. The terms record and sentence may, in fact, be used interchangeably, as a consequence of the definition. The sentence M1 may now be regarded as "referring to" the event in reality as well as to the information. In other words, M1 refers to the fact (reality) and to the message (information)—(as well as to the sentence). Any message may be represented by several distinct sentences or records.

The intended users of the record M1 may conceive of the fact informed about as a composition of two facts which may be referred to by the two sentences:

M2. The customer C ordered article A today.
M3. The quantity ordered in the event M1 was 5 pieces.

Then the fact is a combined fact or a *consolidated fact*. The corresponding message and sentence is then also regarded as a *compound message* and a *consolidated record*, respectively. If the users do not regard the facts represented by the sentences M2 and M3 as compound facts, these facts are regarded as elementary. Then the messages and the records (or sentences) are also elementary. Because we often need these distinct terms, some abbreviations were introduced. Thus the prefix "e-" is used to indicate "elementary" and "c-" is used to indicate "compound" or "consolidated". Thus M1 is a *e-sentence* (or *e-record*) used to represent a *e-message* which is the knowledge of a *e-fact*. The sentence M2 and M3 are *e-sentences* (or *e-records*) representing e-messages about *e-facts*. The e-message represented by the e-sentence M2 may be regarded as information about "customer C", but from another perspective it may instead be regarded as information about "article A" or, indeed, about "today" or about the group (Customer C, Article A, today). The analysis of the c-sentence M1 into the e-sentences M2 and M3 made it clear that the part "5 pieces" of the sentence M1 informs about the event represented by the rest of the sentence (or represented by M2) whereas the part "ordered article A today" informs about the customer C. (This analysis is, of course, essentially a kind of parsing of the natural-language sentence.)

The e-sentence M2 refers to "customer C" as its subject (= entity informed about). The part "customer C, today" i-determines the part "ordered article A", that is, it makes known what information the characteristic "ordered article A" belongs to (or participates in). Similarly, the e-sentence M3 has "event M2" as its subject-plus-time reference and as the i-determinant of the characteristic "the quantity ordered....was 5 pieces".

When we want to refer broadly to both e-records and c-records we use the prefix "i-". Thus e-sentences, or e-records, and c-records are examples of i-records (or sentences). In connection with the term "sentence" it is not necessary to employ the prefix "i-". Analogously M1, M2, and M3 represent a collection of i-messages (or i-propositions). M1, M2, and M3 represent a collection of i-messages (or i-propositions). M1, M2, M3 were formed according to (a subset of) the *rules of formation* of natural language. Other rules of formation were used in

M1a: (Customer, C; Date, D; Order, Article, A) Quantity-ordered, 5 pcs;;
M2a: (Customer, C; Date, D) Order, Article, A;;
M3a: (Event, M2a) Quantity-Ordered, 5 pcs;;

The formation rule here is such as to explicitly reflect the reference structure of the i-messages: the subject reference, followed by the time reference and (then) the property (or relationship) reference. In addition, the important change was introduced that the time reference was made absolute. This, of course, is necessary when the records are to be stored. The brackets are used to enclose the part which provides both subject and time reference. For example, M2 and M2a instantiate two e-record types and may also be seen as representations of two e-message type instances, both of which are the same piece of information. The e-message represented by M2 or M2a is an instance of an "elementary information-kind" or *e-concept* (in this case an elementary predicate). This e-concept may be described by the e-concept schema: (Customer, Date, Order = Article). We may represent the e-message by yet another e-record format which reflects the e-concept:

M2b: ({Customer, Date} Order = Article) (C, D, A);;

In this e-record format the e-concept schema appears in front of the "value tuple" and all e-messages "belonging to" the e-concept may be represented by a set of such tuples, each tuple being coupled to the e-concept schema. (Here the linkage between e-concepts and relations is seen.) In this connection it will be noticed that the i-determination of "ordered article A" by "customer C, today", mentioned above, gives rise to a (multivalued) dependency $(\bar{C}, \bar{D}) \rightarrow\!\!\!\!\rightarrow \bar{A}$, where $\bar{C}, \bar{D}, \bar{A}$ are domains such that $C \in \bar{C}$, $D \in \bar{D}$, $A \in \bar{A}$. The e-concept M2b is seen to give rise to a 4NF relation on its value domains. This is typical of e-concepts. If we introduce the additional e-records

M4a: (Customer, C; Date, D; Order, Article, A) Delivery-Address, Ad;;
M5a: (Customer, C; Date, D) Amount-Due, Amt;;

we find that M1a and M4a have identical subject parts and so have M2a and M5a. We may thus consider, during the data design, consolidating M1a with M4a and M2a with M5a:

M1-4a: {Customer, C; Date, D; Order, Article, A} Quant-Ord, S; Deliv, Addr, Ad;;
M2-5a: {Customer, C; Date, D} Order, Article, A; Amount-Due, Amt;;

or

M1-4b: ({Customer, Date, Order = Article} Quant-Ord, Deliv-Addr) (C, D, A, S, Ad)
M2-5b: ({Customer, Date} Order = Article, Amount-Due) (C, D A, Amt).

These e-records give rise to "e-concepts" (or consolidated i-concepts) as described by the left parantheses. By definition a collection of e-records (or e-records) is an e-file (e-file), provided that the e-records or e-records are of the same type and, hence, associate with the same e-concept or e-concept, respectively.

The e-record M1-4b is a representation of a e-message instance of the e-concept

({Customer, Date, Order = Article} Quant-Ord, Deliv-Addr)

obtained from the e-concepts

({Customer, Date, Order = Article} Quant-Ord)

and

({Customer, Date, Order = Article} Deliv-Addr)

through consolidation on the common subject part

{[Customer, Date, Order = Article]}

This consolidation gives rise to a join on (C, D, A) of the two relations (C, D, A, S) and (C, D, A, Ad). Similarly a e-concept

({Customer, Date}, Order-Article, Amount-Due)

is obtained through consolidation of the two e-concepts having {Customer, Date} as their subject parts. The consolidated (or joined tuples) (C, D, A, S, Ad) and (C, D, A, Amt) belong to two relations that are still 4NF. This is partly due to the fact that the consolidated e-concepts all have identical subjects. Instead, if one would, e.g. consolidate all the e-concepts mentioned above the resulting relation would not even be INF.

INFORMATION, CONTEXT AND SEMANTIC BACKGROUND

Pieces of information, i-messages, are knowledge of particular facts. They must be constructed from references to aspects of these particular facts, as we have seen. But this could only be realized if the receiver of the information has already some background knowledge. Some aspects of this background knowledge are presented below. *Particular* information and *general, pre-existing* background knowledge appear as the two basic components crucial to all information use, communication and data processing.

The need for a general context or semantic background

We stated that any of the e-message representations above could be regarded as complete in a sense. Now we want to make this sense more specific. Let's do this for the e-message format No. (3). To receive the e-message (or to establish it in our mind) as represented by the format No. (3) (Quantity-on-Hand {Article-Type, A; Time, Present; 17 pieces}) we need to have the pre-knowledge of what "Quantity-on-Hand" means as well as what is meant by "Article-Type", by "A", by "Time", by "Present" and by "17 pieces". In addition we must also know the meaning of "Format", "Identifier", "Characteristicum-Type", "Subject Ref.", "Time Ref." and "Value" (Langefors[3, 11]).

It may now be seen that "completeness" of an e-message representation is taken to hold in the sense that those *particular* references are made known that could not be known beforehand. For instance, the particular knowledge that this e-message informs about "Quantity-on-Hand" could not be known in advance. Contrary to this, the general background knowledge of what quantity-on-hand means can and must be known in advance. It is "pre-knowledge" needed for the communication and the usage of the e-message. It is the "user view" for which the data design has to be made.

The result of our discussion so far can be summarized by the statement that the information that may be conveyed by a set of data, D, depends on the person receiving the data in that the semantic background or "receiving structure" S of the person is crucial to interpreting the data. In addition, the time available for the interpretation is also significant. All this can be expressed by a concise "conceptual formula" (Langefors[3]):

I = i(D, S, t)
D = data representing the intended info
S = the "receiving structure" (pre-knowledge) of the user
t = the time available to the user for interpreting the data D
I = the information conveyed by the data D
i = the information function

Another important insight gained from the discussion is:

1a Data *are not* information and
1b Data *do not contain* information
1c But data *may only*, at best, *convey* information to such people the conceptional frame work of whom are both consistent with the data *representation* chosen *and* with the model of the part of reality being involved
1d If the amount of data is large compared to available time, t, no information may come to be conveyed.

A consequence for the information analysis is that in addition to identifying the information needed and defining a data representation of it, (D), one will also have to obtain some knowledge of the "users" infological model S. Alternatively, one will have to verify that the designed data (D) comply with the world view and the language of the intended user.

It may be concluded that once the distinction information/data is recognized it is immediately found that user needs for information and user world views both are to become involved when data are to be designed. But this does not necessarily imply that the user view (S) will have to be represented in the data system, nor does it have to be formally defined. It is only necessary that the data are adapted to the views of the users and that it is properly verified that the users understand the data. If the users understand the data (which can be tested), then this means that their world view has been "involved" correctly in the data design. On the other hand, it may be of interest to represent part of S in their minds. In this way the need for information/data dictionaries and information management became apparent too.

We shall see later that the term "user view of data" is used in recent data base works in a way that is only remotely similar to the concept of "view of the world" or "semantic background" (S) as discussed in the older information systems theory and presented above. We shall try to identify the drawback that is associated with this.

The infological background, or user view, as a system model

It has been seen that the information conveyed by some data depends on the semantic or *infological* framework of the user. It has been stressed that such an infological background must be a system model of (part of) the system in which the fact to be informed about appears (Langefors[11]). Such an infological system model may be in the mind of the user but it may partly be represented by data available to the user on documents or on a computer terminal. This may make it essential to supplement the e-record (e-message representation), which is presented by the system, with the identifier S' of the relevant stored system model. An example may explain this (Langefors[11]) (see also Langefors and Samuelson[12], Fig. 1). For instance, it may not be enough for the receiver of the message to have a dictionary which tells what "Boiler 5" means. He may need a model of the system to see what effect the observed temperature may have and, thus, what decision is called for. On the other hand, any system model can always be constituted by a set of e-message representations.

It is easy to see that "understanding" some data (one e-record for instance) may mean different things. In the minimum case it will imply merely the imagining of the e-fact as the observer perceived it. This would mean that the user of the data would conceive of the part of the world that was perceived by the observer who originated the message. This seems to be the only "understanding" which is generally requested from e-records stored in an IS or data base. It will require a small piece (S1, say) of the semantic background S only. In more demanding cases "understanding" may mean, in addition, a capability of predicting some effect of the observed e-fact.† However, distinct users may be interested in distinct effects. This implies that they all need to possess the "view" S1 while they may then hold distinct additional views, S2 or S3 for instance. It would seem appropriate to regard the "view" S1, basic to all usage of certain data, as the minimum required user view for these data (corresponding to it is the "conceptual Schema" of ANSI[13]) whereas the additional models Si (i = 2, 3, 4 etc.) might rather be regarded as distinct ways of applying the same information I(D, S1, t)). (They correspond to (or, rather, are associated with) the "external Schemas" of ANSI[13]). Now the common use of the term "user view", or "world model" is of the kinds illustrated above by S2, S3, etc., and thus they ought to be regarded as an *application procedure view* rather than as a user view. The distinct applications, associated with S2, S3, etc., will process the same data (use the same pieces of information) in distinct combinations and sequences. This does not necessarily imply that the data *mean* distinct things to the distinct users though these may infer distinct consequences from the same information, because they have distinct problems to solve.

The "view" S1 will typically be associated with objects and object-relations in the real world (the object system) as well as with names and frames of reference associated with these objects and relationships. But notice that this does not mean that all these objects and relationships must be separately modelled in the data base. Rather, whenever a user specifies an e-concept (or e-message type) to be represented in the system he introduces one or more object classes and a property or relation as well. This will thus appear among the data (D) and does not necessarily need to be further represented as data (D(S1) that represent a model S1.

Remark

It should be noticed that the insight that the interpretation of a record (for instance, a sentence) depends on the world view S of the "user" does *not* imply that the view S must be explicitly modelled in a schema, for instance. The only thing that it implies is that any record may only convey information to *some* users and, hence, it is important to ensure that the data (the record) are designed with proper attention given to the views, S, of the intended users. How to verify this is a problem that is open to further research. It should not be treated by tacit assumption.

Information dictionary and information management

An important consequence of the crucial role of the user view (S1 for instance) is that all those who are to share some common data must hold the same (basic) view (S1), at least approximately, whereas they may hold distinct application procedure views (corresponding to what is commonly referred to as "user views"). It follows that it is necessary to manage to have all data terms or names defined in a common, authorized information/data dictionary. This dictionary has to be verified by the relevant user groups and thus it must be made consistent with the relevant user view (such as S1 for

†To an "action oriented" social scientist, inferring the reasons or "motives" behind observed behavior may be the aim, rather than predicting effects.

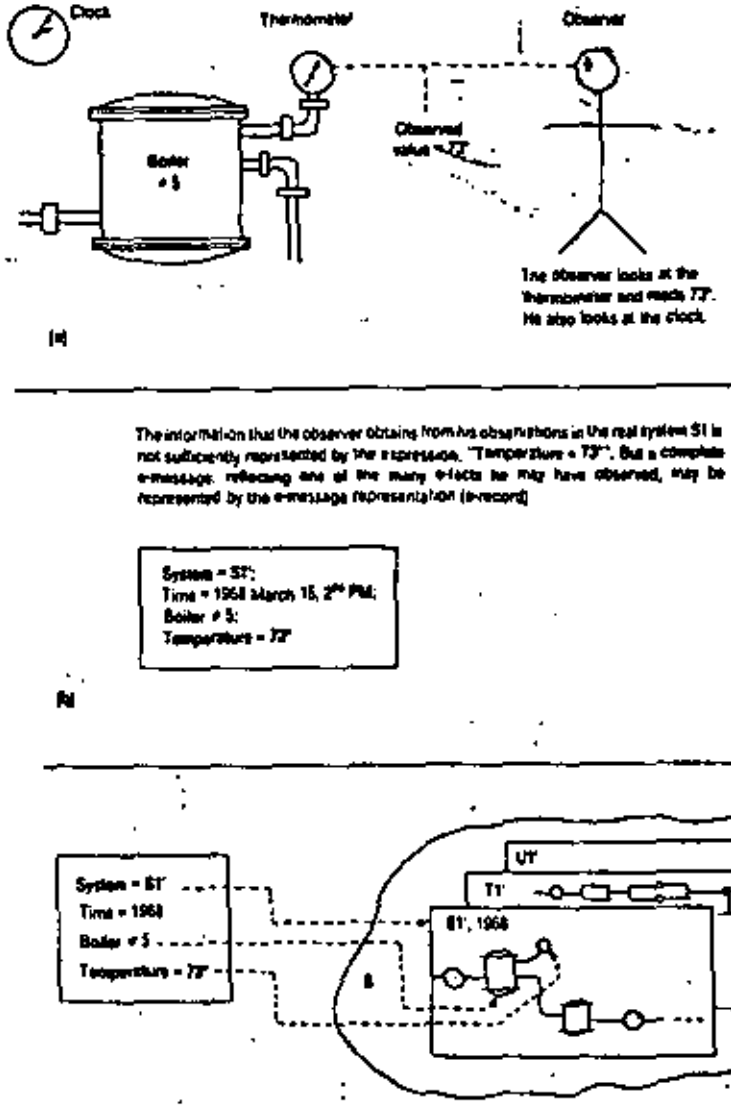


Fig. 1.

instance). Consequently, any time that some e-message type is to be documented as required information, all the names involved must conform to the dictionary. However, it will often happen that new names are needed, when new e-message types are introduced. It will then be necessary to have institutionalized a managerial procedure for how to introduce and authorize new names. These names will, typically, be names of object classes or property types. Thus it is seen to be necessary for any IS design process to have a well established information management procedure (Langefors[14]). This need for data management and information management functions has also been recognized, more recently, by (CODAYL[15]) and (ANSI-SPARC[13]). It is desirable to have the information management function supported by a computer (Langefors[14]). Of course, the information dictionary will have to contain entries not only for object classes and property types but also for e-concepts and other i-concepts. Furthermore, not only will the meaning of each name have to be documented but various qualitative and quantitative characteristics will be put down and serve as a basis for data and process design as well as for integrity constraints.

Purposive information background

We have discussed how the understanding of data, receiving the information, depends on the personal world views, S. This is a kind of cognitive background—what the data users know of the world. But the information received will be used to support decisions and actions. This, however, does not only involve factual information but also goal or purpose information. In management information system design one is interested in providing goal information, in addition to the factual information, in order to influence or control the decisions or actions. However, as shown in (Langefors[24]) decisions made by people are necessarily dependent on their personal "purposive inclination". Thus, human beings have internalized goals that they are usually unaware of and that

automatically influence their decisions. So decisions cannot be totally controlled by explicit goal information and the expected behavior of data users is thus not only dependent on their cognitive world view (S) but also on their "goal views", or intentions. This is testified, for instance by the fact that people normally make decisions without making explicit the underlying goals or aims and, even, without being aware of them.

The reference parts of an e-message and the infological background

We stressed that the reference parts of an e-message must *make known* what they refer to but that this may be done in various ways. We now may infer that how it may be made known will depend on the structure of the real system and on the infological system model (S) available to the users. We may illustrate this fact in connection with the earlier example

eml: Quantity-on-Hand (Article-Type, A; Time, Present; 17 pieces)

First, the time reference can obviously not make known the time it refers to if the e-message representation eml would be stored without supplementary time reference data, in a data base.

Secondly, the Subject reference Article-Type, A; does not make known, by itself, the object it refers to. First we notice that we probably have to do with an object which is, itself, a collection of objects that are articles of the type A. Furthermore we guess that this collection is stored somewhere. Suppose this collection is stored in Store No. (1) and that such articles may also be stored in Store No. (2). Now we must conclude that the expression "Article-Type, A" does not, alone, make known that which was intended. It must be supplemented to "Article-Type, A; Store, No. 1;" and the infological model of the users must be compatible with this.

You may now ask whether the needed supplementing should be done by the analysts or by the users. Obviously, the analysts could only do this if they know that there is more than one store. Regardless of who does the specification, if it has been verified by the relevant users that the object reference chosen for the e-message formulation does identify the intended object, then the data can be designed. It is not required that the data system has a stored system model (if the data system is not an artificial intelligence system). The data system will associate data objects with the object identifier chosen and the users will associate this identifier with the real life object as defined by themselves.

SOME DIFFERENT ASPECTS OF USER VIEWS

The information that may be conveyed by data depends on the view of the world, held by the users of the data, as well as it depends on the language that these people use to express the views. This is one of the basic points-of-departure of infology and information systems theory but, also, it is a focal point of hermeneutics, the "science of understanding". We present some aspects provided by these fields of study. More data system-oriented studies such as data base theory and structured-programming theory speak of "user views of the data" and thereby consider how the same data are processed distinctly in distinct applications. The aim of this paper has been to compare—very briefly—these distinct perspectives of "views".

Abstract IS, real-world model and user view

The e-messages correspond to e-facts in the world. Hence the abstract IS (AIS), consisting of all e-messages, e-algorithms and e-concepts, forms a model of a selected part of reality. This was the view utilized in early IS-theory (Langefors[1,3]) and it still is a basic infological view (Langefors and Sundgren[16]). This view stressed, as we have seen, that the abstract IS (or infological model) (AIS) must always be based on another world model, the semantical background (or collection of personal world views) S.

The abstract IS, AIS, will be assumed represented partly by data (D) and programs, in the typical IS design case. Contrariwise, the personal world view collection, S, will not be represented through data in the typical case. It is only necessary that it exists in the minds of the IS users or, more precisely, that relevant parts of it are invoked at relevant times.

The data base literature often ignores the term "information", in its meaning of knowledge as used here. Instead one speaks of the data base and of a "real-world model" assumed to be represented by the data base. It should be observed that the "real-world model" of the data base literature corresponds to our abstract IS (AIS) and not to the "semantic background system" S. Because the distinction AIS/S is never articulated in the DB literature it is usually quite confused as to what should be represented in the data base. As a consequence, it is often taken for granted that all of the real-world model (thus both AIS and S) ought to be put into the data base. This is obviously both undesirable and impossible. Behind any representation of a world view, such as S, there will always have to be another world view S'. Thus, regardless of how much is put inside the data base, there must always be a worldview outside it, available to the users.

Changing personal views, a concept of concepts

A special problem with the importance of user views for the interpretation of data is that personal views change continually. This is sometimes intended, sometimes not. In fact the IS itself may well contribute to this change. As a result, any IS will tend to loose some of its relevance over time.

One example of an intended change of view would be the possible effect that the efforts on reshaping the usage of the term "user view" might have upon some readers of this paper.

The phenomenon of concept formation and change is, of course, a profound psychological phenomenon. However, as shown in (Langefors[17]) it is possible to explain much of it based solely on simple data modelling

and assuming the saving of memory space to be the main criterion guiding the concept forming process.

To illustrate, we bring an extremely simple example. Suppose we have an IS which receives and stores information as represented by e-records such as these:

TOM has a head;
NAUTILUS has a hull;
TOM has a body;
BILL has a head;
HARRY has an arm;
TOM weighs 150 pounds;
BILL has 2 legs;
etc.

The IS compiles the e-records into internal data structures as depicted in Fig. 2. These structures will be referred to as data objects. A pattern analysis algorithm operates on these data objects and extracts structures that are found to be common to several data objects. Thus new data objects get defined and names are assigned to them. They will be referred to as *concepts*. From the data base associated with Fig. 2, a concept data object called Person is obtained from the algorithm, Fig. 3.

Notice that the concept object, Fig. 3, has one constant or closed, part named Person-Basic and one open part called Person-Form. The closed part, Person-Basic, contains a set of complete, or closed, property references (such as "arms, qty, 2;") whereas the open part contains names of property types but leaves the property values open (such as "weight, pounds ();"). The open concept data structure part allows real-life objects (called persons in this case) to be regarded as instances of a concept even though they have distinct values for those properties listed in the open part. (For instance, persons may have distinct weights, nevertheless they all have the common characteristics of having some weight.)

Of course, the open property values will usually be constrained to stay within specified bounds (for instance, persons must have weight values between zero and 400 pounds). Such defined bounds serve as integrity constraints for the data base.

It is interesting to see that already now we may recognize some properties of the concept data objects that have significance for IS design. Thus, the concept objects (such as in Fig. 3) have a degree of permanence which makes it possible for people to keep them avail-

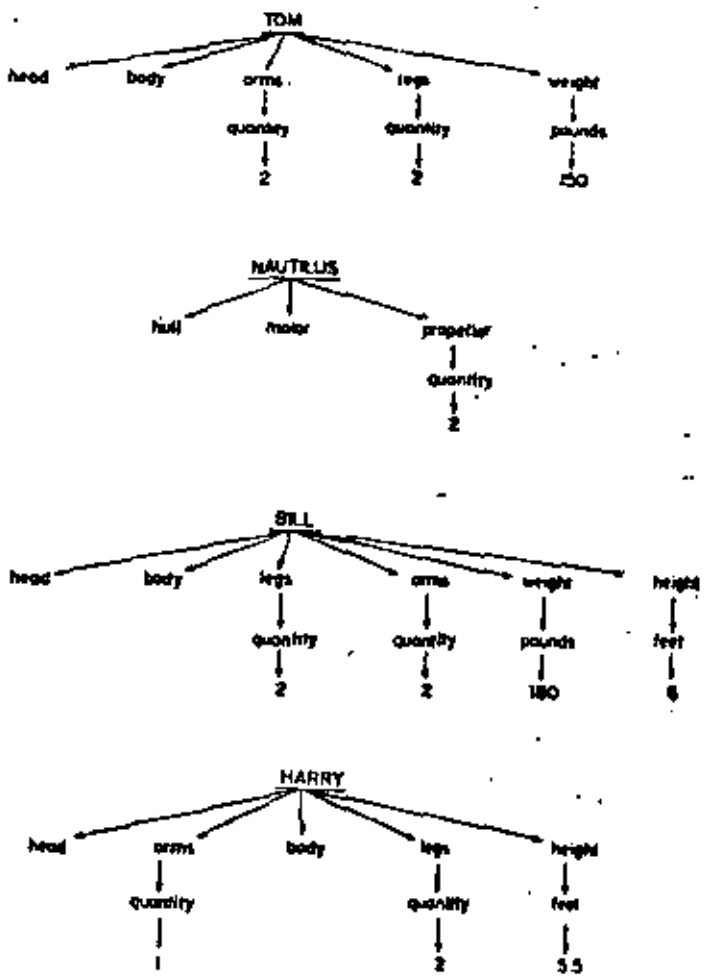


Fig. 2

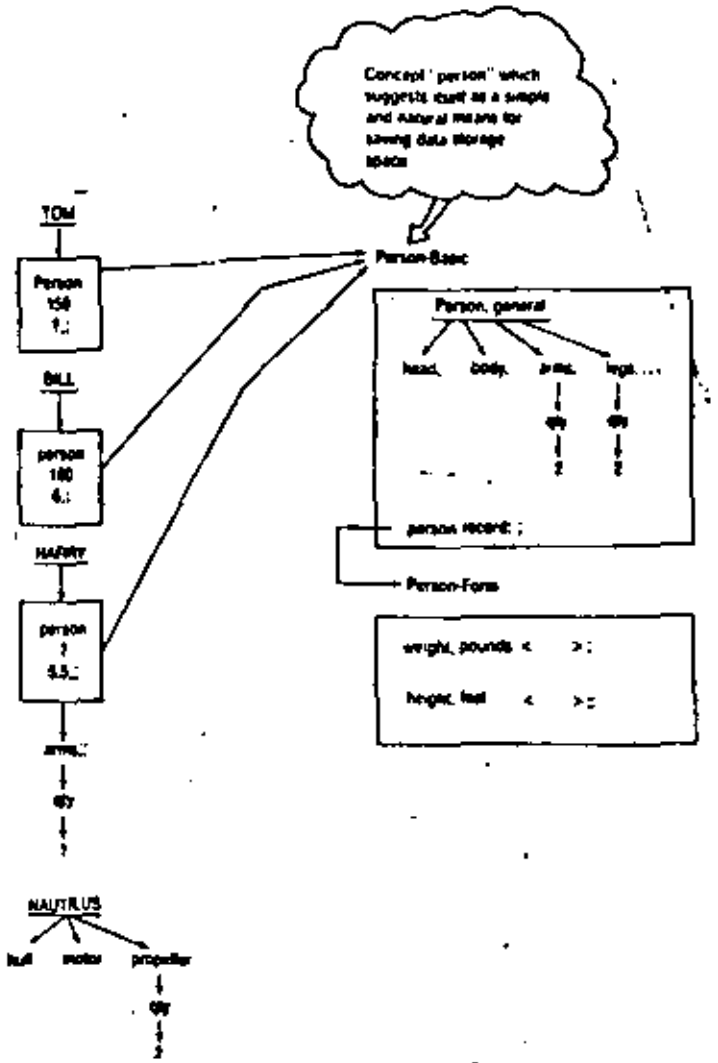


Fig. 3.

able. They are obviously closely associated with the "semantic background" or "user view" S as we discussed earlier. Instead, the open value positions represent information that cannot usually be available to people in advance. It is that information which calls for particular messages and, hence, needs data records (such as sentences) for its communication. Furthermore we now also see what *particular context* is needed with any single term before it can convey information. This particular context, we can now see, will have to identify the concept object that the value belongs to as well as the proper place of the value in that concept object. We may now also conclude that the open parts of concept objects correspond, roughly, to data record types in data bases. Finally, the names, or terms, appearing in a concept object must be names of other concept objects that are available to the information users, in their minds or through some external means.

The concept objects, clearly, form semantic systems or networks because the property references that they contain refer to other concept objects.

The introduction of the "concept-data, objects" may be justified solely by the saving of memory space (in human memory or in computer) that they bring about. For instance, the data object Tom of Fig. 2 which we may write as

Tom (head; body; arms, qty, 2; legs, qty, 2; weight, pounds, 150);; may be substituted by

Tom (person; 150;?);;

if it is assumed that the value terms following a concept name, belong to the open value slots of the open part of the concept object and "?" is used to denote a missing value specification (which thus leaves a value still open).

In general we will encounter data objects that are at most partly consistent with a concept object. The difference may be that the object has another value for a property than one specified in the closed part of the concept. For example, the data object Harry, in Fig. 2, specifies one arm only. Alternatively: the object may be lacking a property reference altogether or it may contain property references not contained in the concept object. For such object a reference to a concept may still be used if those (exception) property references that deviate

from the concept structure or content are added. For instance the object Harry in Fig. 2, may be written

Harry (person; ?; 5.5) arm, qty = 1

To decide whether a deviating data object should be regarded as associated with a concept (while supplemented with exception data) or not, one may assess the resulting data space requirement.

For any given set of data objects—any given data base—and any established concept object, an algorithm may identify all those data objects that may become reduced by referring to the concept (as well as those being increased). The total saving may then be computed. Now this may be repeated for a number of modified concept objects so as to find the concept object which is most efficient in reducing memory space. It is clear that this procedure will not have a unique result so that some strategy for choosing one solution out of a set of equally efficient solutions will have to be created.

Obviously, when a number of concept objects have been created one may apply the same procedure again to create concepts that reduce groups of concepts. Thus a hierarchic concept structure will emerge. The model presented may be seen as a possible, simplified model for how human views of the world are developed.

The concept of "concepts" based on memory saving only, appears to be something much more simplistic than "concepts" as commonly thought of. However, several known characteristics of human conceptions are reflected by the model, simple though it is. Hence it is of interest to look at some statements that may be derived from the model:

(1) The definition of a concept will not be unique, several concepts may have approximately equal efficiency.

(2) Whether a certain object should belong to the extension of a certain concept or not, is a question which may not have a unique answer—hence concept logic seems to be three-valued (true, false, not decidable). It is of interest to notice that this is similar to the answer possibilities of statistical decision theory.

(3) Concepts will change as the subject will experience new perceptions.

(4) Distinct persons may generate distinct conceptions even when they have the same experiences.

(5) There is no fixed set of properties that all instances must possess, rather, any instance must have enough many properties in common with the concept.

It seems likely that these five statements, suggested by the very simple model, would be still more relevant the more complexity is added to the model. Hence the statements appear to be generally true. All these conclusions are clearly consistent with experience and thus this experience may be explained by the simple assumption of a storage economy criterion. It seems also clear that this model gives a lot of guidance for the consideration of user views and user conceptions in IS design and operation. It is worth noticing, especially, the indication of concept variation with time that has to be reckoned with, according to the model. Any system containing automatic concept forming mechanisms will change its set of concepts continually, as new information is received.

A famous work which is concerned with some aspects of concept changes and, hence, "user views" and their importance is (Kuhn[18]). Kuhn uses the term "paradigm" in several ways, some of which are related to effects of varying infological/conceptual "user views".

Hermeneutics, infology and "user views"

Infology and IS design are concerned with how data may be related to the world and to information about the world. Hermeneutics is a field of inquiry which is explicitly devoted to how human beings can understand or interprete behavior or texts such as historical or law texts, for instance. Thus infology and hermeneutics clearly have common interests. Hermeneutical work is usually presented in a verbose and entangled prose. This often makes it difficult to know whether anything can be contributed to IS design and, indeed, it is natural to a scientist to question whether hermeneutical texts carry any precise meaning at all. But once the close relation between IS-analysis and hermeneutics is recognized, it becomes desirable to seriously look into hermeneutics. Now hermeneutics itself is perhaps more concerned with injecting meaning into a text (Sinngebung) than with extracting meaning out of it (Auslegung). This of course, implies that the question about precise meaning becomes somewhat irrelevant. Thus, in reading hermeneutical works one should try to see if it is possible to make sense out of them through explicit, constructive efforts of interpretation. In any case, it is clear that the question of user world views and data understanding may be regarded as a hermeneutical question. For this reason it appeared desirable to try to see if it was possible to attach concrete, infological meaning to selected sections of hermeneutical texts and to try to find out whether that field might be useful to IS design. This little study turned out to be reasonably successful (Langefors[19]).

Hermeneutics stresses the critical importance of the "pre-understanding" (Vorverständnis) held by the subject. A typical hermeneutical view is the one (by Gadamer) which claims that interpretation is not merely a looking for an objective content in the text (which may never have existed), nor is it merely a looking for the true intentions of the author (which he, himself, may not have been aware of). It is rather something "belonging to the current event of interpretation" and of the "being-in-the-world" of the interpreter. It is certainly not easy for a scientist to attain a Gadamer kind of "being-in-the-world" and to understand in that sense, but however that may be, one is clearly concerned here with "user views" of some kind.

For a more penetrating discussion let's pick up another piece of hermeneutical text and try to attach infological meaning to it (Langefors[19]). The text is from (Apel[20]) (my translation from German).

> "A language-hermeneutical analysis assumes that the understandable human behavior responses contain, themselves, the property of understanding, being, as they are, language related intentional images. This

analysis has to conclude that the world knowledge (Weltdaten), against which the behavior is to be understood, must itself be understood against the understanding of this same behavior".

This piece of text may, perhaps, be watering the mouth of a recursion-loving computer scientist but to "unbundle" this spaghetti phraseology does not appear to be easy. But this is what should be tried.

To bring the problem within the grasp of infology we may start by assuming the simplest possible case when the human behavior to be understood is a single e-fact (an "elementary event" in this case) so that the understanding would give rise to an e-message ε. The observed behavior may now be regarded as the data ε̃ representing the e-message ε. We may now put down the infological equation

$$\varepsilon = i(\tilde{\varepsilon}, S, t)$$

which says that the information (understanding) ε will be obtained through a process $i(\tilde{\varepsilon}, S, t)$ which works on the data ε̃, and employs the pre-understanding S during a period of time t. Now, once the knowledge ε has been generated, the pre-understanding by the subject may change to $S' = S + \varepsilon$. This symbolization is, of course, merely to be understood in an intuitive sense but it is not hard to imagine how it could even be modelled in a computer, and thus be made very concrete and formal.

We are now led to the new equation

$$\varepsilon' = i(\tilde{\varepsilon}, S + \varepsilon, t)$$

which indicates that the knowledge ε'—which is generated by a reiteration of the same process (i) from the same observation (ε̃) but with the new knowledge background $(S + \varepsilon)$—might be different from ε. This corresponds, possibly, to the text "...must itself be understood against the understanding of this same behavior". However, if we imagine a computer simulation of the equation $\varepsilon = i(\tilde{\varepsilon}, S, t)$—which we do in order to try to test our understanding in a concrete way—then we would think of S as being represented by a network data structure, $\tilde{S}$ (e.g. a semantic network), and the understanding process (i) would establish an integration of ε̃ with $\tilde{S}$. Thus ε would be represented through $\tilde{S} + \tilde{\varepsilon}$. Then it would be natural to a computer scientist to assume that the model would be such that the repetition of the process (i): $\varepsilon' = i(\tilde{\varepsilon}, \tilde{S} + \tilde{\varepsilon}, t)$ would change nothing at all: that is he might assume $i(\tilde{\varepsilon}, \tilde{S}, t) = i(\tilde{\varepsilon}, \tilde{S} + \tilde{\varepsilon}, t)$. However, it is easy to conceive of a change of the above model. In fact, the equation itself suggests a more complex model because the effect of time for interpretation (t) ought to be represented also. A modification which suggests itself is to assume $\tilde{S}$ to be stored on an auxiliary store and to be searched by the process (i) which brings relevant parts of it into main memory (corresponding to areas of higher awareness in the human memory). Thus, the time constraint (t) will restrict the time allowed for search in the backing store. In this modified model the part of $\tilde{S} + \tilde{\varepsilon}$ that resides in main memory may well be distinct from (and more powerful than) that which resulted in the first execution of (i). Correspondingly, ε' would be distinct from ε and would be associated with a semantic network, more powerful than the previous one with respect to the understanding of ε̃. Now, clearly, the modified model does *construct* sense to the piece of text by Apel that we are discussing, though it is doubtful whether this constructed meaning was intended by the author (Apel). Nevertheless, our hermeneutical exercise appears to have been quite successful in generating some deeper understanding (as well as suggesting some interesting research of an artificial intelligence kind).

The phenomenon that we have just discussed is, incidentally, a simple example of what is sometimes referred to as the "*hermeneutical circle*".

The interpretation experiments made above indicate, through their positive results, that hermeneutical texts should be given some extensive studies by information analysts. The authors of such texts may not be aiming at a very concrete, engineering type of understanding. Nevertheless, such a concrete understanding was easily possible to reach in the above interpretation experiments. Furthermore, this became possible through explicit hermeneutical efforts of constructing meaning that did fit in this attempt. The outcome of the efforts was a deeper understanding of the process of interpretation. It seems reasonable to expect that when the small extracts of hermeneutical texts, used in the experiments, turned out to be quite helpful, more useful knowledge should be obtained from more extensive studies. Because hermenutics is very much a matter of "views" of authors and their readers it appears to be potentially advantageous for more extensive studies of "user views" as related to the use of data in information systems.

Process-oriented views on data and user views on data

It was pointed out in (Langefors[3], Section 212.2) that it is advisable to

> "make use of the fact that the record is uniquely defined *to some extent by file and to some extent by the* actual process. Thus one should have a *basic record description* which is defined by the file and an additional structural description which adds.... to the basic record description. Only the latter would then have to be defined for the process".

This idea was also forwarded in [Olle[25]] and has much more recently been taken up in the data base management systems work. For instance the "basic record description" concept in (Langefors[3]) is very closely related (if not identical) to the "physical structure" or to the "storage schema" of the CODASYL DATA DESCRIPTION LANGUAGE COMMITTEE 1978 (Metaxides[21]) and other CODASYL reports during the 1970s. Likewise, there is a clear resemblance between the "additional, structural description", mentioned above, and "logical structure" or the "schema" (and the subschemas) of CODASYL. An alternative term to "subschema", proposed by (ANSI/SPARC[13]) is the "external schema" and the associated "external level" of the data base architecture. In the ANSI/SPARC terminology the "basic record description", mentioned above, corresponds to the "internal level". The "external

level" is said to be concerned with "individual user views" (Date[22]). It is to be noticed that one is, here, concerned with user views of the data base, not with user world views or infological user views though, unfortunately, this distinction is usually not made clear.

The common use of the term "user view" in the data base literature employs the idea that the data base is modelling part of the world and that an individual user sees only a subpart of this part of the world. Consequently, the user would only "see" a part of the data base such that of all the data about a real-world entity, contained in the data base, an individual user will only see a subset of the data base objects and only a subset of the properties of each of these objects. This may suggest the term "user view". However, it seems important to recognize that this is only a very special aspect of the user's view of the world. It ignores entirely the infological insight that the information—which the users may obtain from the data—is totally dependent on the general "semantic background" of the users, as we have seen. It takes for granted that if distinct people are concerned with distinct parts of the world then each of them would be served by obtaining those data from the data base that are associated (by somebody else) with those parts of the world.

The way the term "user view" is employed in recent data base literature may be further illustrated by presenting some quotations from Date[22]. "The external schema defines the user's view of the data base". The data orientation—rather than information orientation—of this conception of "user view" is further exhibited by the following two quotes from Date[22]. "Suppose that the user wants to see the same information as in the PARTLOC example, but in the form of a hierarchy rather than a relation" (p. 182). "For instance, in the conceptual schema we may have

DOMAIN WEIGHT *NUMERIC* (4)

whereas the corresponding domain in an external schema to be used in a PL/I application may be defined as

DOMAIN WEIGHT, *FIXED BINARY* (14)".

It is clear that the "user's view of the data base" which is thought of in these pieces of text—and several similar ones in data base literature—is very different from what the term "user view" would suggest from an infological perspective.

In Codd[10] there is another illustration of the data oriented way in which "views of the data" are in fact treated in data base works: "For application programs that do more than merely read the data, there are theoretical limitations which must be observed if the data base integrity (including consistency of all permitted views) is to be maintained". As we shall see, the "theoretical limitations" considered are merely those of a purely formal kind. No theoretical limitations of a psycho-linguistic-conceptual (or infological) kind are, at all, mentioned. Furthermore, it seems to be assumed that "all permitted views" could be consistent. This is a very strong statement to make. And our present discussion indicates that it is not tenable.

In the illustration Codd assumes that "the community schema" contains the relations R(A, B) and S(B, C), which at a certain moment have the tabulations:

R(A B)		S(B C)
sl		lu
tl	and	lv
—		—

and that a user requests the schema T(A, B, C) where T is the natural join of R with S on the common attribute B. The tabulation of T becomes:

T(A B C)
slu
slv
tlu
tlv
—

Now the assumption is made that the user wants to delete the triple (t, l, v) and it is pointed out that if he was allowed to do this T would change to a relation that is not the join of *any* two relations.

Clearly, there is an implied assumption here that the user is concerned with how the data are stored—or how they might be conceived of as being stored. This may be important to a user who is an application programmer. From an infological point-of-view the problem illustrated would be irrelevant (similarly to the above illustration from Date[22]. From the infological perspective the data (e-records) in the data base are thought of as *representing* information entities (elementary messages) that are instances of elementary information kinds (e-concepts). The user expects the data base to contain e-records of many distinct kinds and that he will obtain a proper selection of them on request. He does not require that the other data (e-records) appear as non-existent and he wants to disregard the way the data are aggregated (into n-ary relations or hierarchies or whatever). Thus, for instance the relations R(A, B) and S(B, C) might be representations of e-concepts and the (infological) user would expect the management of these to be unaffected by the request of some (other) user to be allowed to forget a certain tuple (such as t, l, v) or a combination of e-records. Furthermore, if the user wants the tuple (t, l, v) he wouldn't care whether this is a tuple from T or is combined from e-records from R and S. He probably also would not want to learn about the name T and the content of T. On the other hand, the problems discussed in the illustration are real problems at the stage when the "implementation" is about to be designed.

It may be of interest to look also on an example of how, in the structured-programming literature, the "problem environment and.... our understanding of it" is regarded as defining the structure of the data while, then, a particular processing structure is defined (Jackson[23]). We quote:

"(i) consider the problem environment and record our

understanding of it by defining structures for the data to be processed;

(ii) form a program structure based on the data structures".

It appears that the understanding of the problem environment mentioned, is related to the data base text's concepts of "user views". Then, when it is suggested to form a program structure based on the data structure, this indicates that the author has already imposed some thoughts about a specific processing task upon his "data structures". But such a structure may not at all be inherent in the structure of the information (or the reality represented by the data). For instance records on store movements are from one point-of-view independent of each other; each represents an individual event of issuing or receiving a set of parts. When it is decided to sort the records into part-number order and to compute distinct summaries, this means to impose upon the data some structure which reflects what one wishes to do with the data. It is then not surprising that this imposed, process-oriented, structure of the data can be used to determine the structure of the program for this specific application. But, the same data may be processed for another problem after imposing quite another structure. Clearly, these structures are not inherent in the information represented by the data, or in the reality informed about. In fact, the typical examples of the use of the basic structural categories: sequence, iteration and selection reflect decisions on how to navigate across some data in order to solve some specific problem and they have little to do with the infological structure underlying the data themselves. Thus the view of the data and their relations to reality is rather distinct from the infological (or conceptual view). It seems, rather, closely related to the "user view of the data" as the term is employed in the data base literature. It appears that the structured-programming would become more systematic—more structured—if one would define the infological structures, independently of the intended processing first, before proceeding to the design of the processing-oriented data structures that one does not start with.

A more general and abstract process-oriented view of the information structure (thus not merely the data structure) was developed in Langefors[1, 3]. Information precedence relations were analyzed by identifying the information units (precedents) that may be used to derive specified information units. This precedence structure was used both in order to making the information needs analysis systematic and to finding the record and file consolidations and process groupings that would be efficient from the data transport (access & transfer) point-of-view. Thus data design and program design was also made systematic by information precedence analysis. Information precedence graphs, content graphs (component graphs) as well as lists and matrices were used as tools for describing the information precedence structure.

CONCLUSION

For some years the author's ideas of information systems or "large shared data bases" have been that all data in the data base should be available to all "legal" users. To this end all data should be adapted to what is usually called a "community view". According to recent data base writings, the data administrator is supposed to define this community view. This is to be done by means of the conceptual schema (that is the infological model). Then each particular user view is assumed to be serviced by a subset of the data, that is arranged with consideration to the processing pattern assumed to be associated with the particular user view. This data subset and arrangement is supposed to be declared by an external schema, presumed to model the particular user view. All data in the data base that are not within the scope of an external schema are assumed to be invisible—indeed non-existent—to the particular users associated with the particular external schema.

The infological (or conceptual) aspects of data and information—as well as of "user views"—that has been discussed in this article imply some fundamental disagreement with the current data base theory aspect (or "datalogical aspect") of user views, community view and external schema. Below four aspects are described:

(1) If the data to be used by a particular user have been modified in order to be consistent with some kind of "community view", they may be unintelligible to the users, no matter what data selection and rearrangement is brought about by the use of the external schema. In other words, *it may be impossible to make some data "shareable"*.

(2) The user's view of data depends on his view of the world. To such a view one or more sets of data may be adapted. The user may then want to use the same data for the solution of distinct tasks. *Based on the same view of the data* he may thus want to *process* them in distinct ways and, for that reason, he may want to arrange them differently. This may call for distinct externals schemas that, however, have to be based on the same user view. *The interpretation of some data must always be based on the same* (infological/conceptual) *user view—the view for which the data have been designed.* But these data may be rearranged for processing reasons and the application programmer may view the data from the processing point of view. Thus, the *programmer* may view the data according to *how* they are to be processed—rather than according to what they *mean*. But the data must always mean the same while distinct inferences are drawn from them in distinct applications.

(3) The idea of *one* community view, declared by *one* conceptual or infological schema has to be replaced by a system of conceptual (or infological) schemas. One or more of these schemas, infological/conceptual subschemas, may describe such information as has been possible to establish as "community" information. This cannot be decided by the "data base administrator". It must be determined through learning and negotiation among the relevant users. Some other infological/conceptual subschemas may describe information that can be shared by distinct user groups but that requires distinct frames-of-reference for distinct user groups. This assumes that the users have "distinct but reconcilable" infological views. The implication here is that distinct

users will require *distinct data* to obtain the same information (approximately). This is the case of "shareable information" but non-shareable data. Finally there will be infological/conceptual subschemas declaring user views that are irreconcilable and that, hence, correspond to non-shareable data.

(4) The hypothesis—forwarded above—that there will be user views that are incompatible, implies that the information systems or data bases will contain "islands" of non-shareable data. One consequence is that it will also be pointless to try to cater for formal consistency testing among distinct subsystems. Recognizing this fact (if it is a fact) will save a lot of useless analysis, formalization, and verification work as well as a lot of gathering of testing data. "Total data base integrity" will be meaningless and impossible, while "island-wise" integrety will still be important—and easier to achieve.

REFERENCES

[1] B. Langefors: *Some Approaches to the Theory of Information System*, BIT 1963.

[2] B. Langefors: Towards integration.....In *Vistas in Information Handling, Augmentation of Man Intellect by Machine*. (Ed. by P. Howerton). Spartan Books, New York (1963).

[3] B. Langefors: *Theoretical Analysis of Information Systems*. Studentlitteratur, Lund, Sweden, 1973 Edn. with Auerbachs and Petrocelli/Charter, New York (1966).

[4] Y. Bar-Hillel: Language and information. *Selected Essays on their Theory and Application*. Addison-Wesley, Reading, Mass. (1964).

[5] R. Carnap: *Meaning and Necessity*. The University of Chicago Press (1970).

[6] J. Piaget: *The Origin of Intelligence in the Child*. Penguin Books, New York (1936).

[7] Wittgenstein: *Tractatus Logico—Philosophicus*. Routledge and Kegan Paul (1921). 2nd Impression 1963.

[8] Bower and Anderson: *Human Associative Memory*. Wiley, New York (1974).

[9] N. Lyons (Ed.): *New Horizons in Linguistics*. Penguin Books, New York (1970).

[10] E. F. Codd: Recent investigations in relational data base systems. *Information Processing 74 (IFIP Cong. 1974, Stockholm)*. North Holland, Amsterdam (1974).

[11] B. Langefors: *Introduktion till informations-behandling* (Introduction to Informatics, Swedish). Natur och Kultur, Stockholm Sweden, 1968.

[12] B. Langefors and K. Samuelson: *Information and Data in Systems*. Petrocelli/Charter, New York (1976).

[13] ANSI-SPARC: Study group on data base management systems. *Interim Rep. FDT (Bull. of ACM-SIGMOD)* 7, No. 2, 1975.

[14] B. Langefors: Some problems with recognizing and identifying similar info. Types in EDB systems. *IB-ADB* 1967, Dept. Adm. Data Proc. Institute of Technology, Stockholm (also in *Systemering* 70, pp. 83 FF. Studentlitteratur, Lund, Sweden).

[15] CODASYL: Systems Committee. Feature analysis of generalized data base management systems. *Tech. Rep.* (May 1971).

[16] B. Langefors and B. Sundgren: *Information Systems Architecture*. Petrocelli/Charter, New York (1975).

[17] B. Langefors: A concept of concepts. *ID-ADB* 1070, No. 29, Dept. Adm. Data Proc. Royal Institute of Technology, Stockholm, 1970.

[18] T. S. Kuhn: *The Structure of Scientific Revolutions*. The University of Chicago Press 2nd Edn. Enlarged (1970).

[19] B. Langefors: Hermeneutics, infology and information systems. *TRITA-IBADB 1052*, 1977. Royal Institute of Technology, Stockholm. Dept. Adm. Data Processing.

[20] K.-O. Apel: Szientifik, Hermeneutik, Ideologie-Kritik. In Man and World. *Int. Philosophical Rev.* 1, 37–63 (1968).

[21] A. Metaxides: Report of the CODASYL data description language committee. Special Issue of *Inform. Systems*, Vol. 3(4), 248 (1978).

[22] C. J. Date: *An Introduction to Data Base Systems* 2nd Edn. Addison-Wesley, Reading, Mass. (1977).

[23] M. A. Jackson: *Principles of Program Design*. Academic Press, London (1975).

[24] B. Langefors: *System för Företagsstyrning* (Systems for Management Control, Swedish). Studentlitteratur, Lund, Sweden, 1968.

[25] T. William Olle: *Data structures and storage structures*. In *File Organization*. IFIP Administrative Data Processing Group (IAG) Swets & Zeitlinger, Amsterdam (1969).

DISEÑO DE BASE DE DATOS

FURTHER ANALYSIS OF THE ENTITY-RELATIONSHIP APPROACH TO DATABASE DESIGN

EXPOSITOR:
ING. DANIEL RIOS ZERTUCHE

MAYO, 1984

Further Analysis of the Entity-Relationship Approach to Database Design

PETER A. NG

Abstract—The nondeterministic or deterministic entity-relationship model of a database is formally defined as a user's view of that database in terms of a collection of time-varying relations: the regular or weak entity relations, or the regular or weak relationship relations. Both nondeterministic and deterministic entity-relationship models have the same strength to characterize information concerning entities and relationships which exist in our minds. An improved table form of the relations is introduced to provide a clear and concise user's view of databases. The basic concept of the entity-relationship approach to the logical database design is provided, and is used to derive *3NF* relations. Finally, a method of representing physically these relations, which are generated by the use of the entity-relationship approach to the logical database design, is presented. Thus, the entity-relationship approach to the logical and physical database design can also be realized.

Index Terms—Determinism and nondeterminism, entity-relationship approach, functional dependency, logical and physical database design, normal forms, relational, hierarchical, and network models.

I. Introduction

A DATA model, called the entity-relationship model, has been proposed in [2], [3]. And the recent development of its use for systems analysis and design has been summarized in [15].

In the entity-relationship model, one of the major characteristics is the existence of attributes, each mapped from an entity set or a relationship set into a value set or a Cartesian product of value sets. In this paper, we shall describe formally that a deterministic entity-relationship model is a model, in which each of the attributes is a many-to-one mapping. If an attribute is a many-to-many mapping, then it is a nondeterministic model. The nondeterministic and deterministic entity-relationship models are defined in terms of entity-relationship relations or relations, each described by a regular or weak entity relation, or a regular or weak relationship relation, according to the different levels of logical views of data [2], [14]. It can be shown that both nondeterministic and deterministic models have the same strength to characterize the same information about the real world.

The entity-relationship approach to logical database design, described in this paper, consists of three major phases: 1) defining the enterprise schema using the entity-relationship diagram that conforms to the description of a database application, 2) translating the enterprise schema into an enterprise-user schema, called the entity-relationship relations, such that the conversion of nondeterministic relations into deterministic relations can be done with ease, and 3) transforming the enterprise-user schema (or directly from the enterprise schema) into a user schema. The user schema can be expressed by a collection of regular or weak entity relations, or regular or weak relationship relations. These relations in an improved table form have a clear and concise description of the user views of the database. Indeed, the user schema also can be expressed by the network, hierarchical, or relational diagrams [3]. And the translation process from the entity-relationship diagram to the hierarchical diagram has recently been studied [13] to provide a unified approach to the logical design of the hierarchical model.

In defining the enterprise schema in terms of an entity-relationship diagram, one of the major concerns is that whether the diagram conforms to the description of application for an enterprise [11], [12]. The other concern is that whether the relations, obtained by translating from the entity-relationship diagram, are "equivalently" corresponding to *3NF* relations of the relational model. Within the realm of these concerns, this paper outlines the major steps in logical database design using the entity-relationship approach.

In the course of constructing the physical representation of an entity-relationship model, a disciplined data-structure diagram [2] is obtained by translating from the entity-relationship diagram. It has been shown that this data-structure diagram can be used to recognize the multivalued dependence [11]. In addition to the entity-relationship diagram, this data-structure diagram and the user schema can be used as major components of an integrated design tool to assist in the process of the physical database design. Thus, the entity-relationship approach to the logical and physical database design can be realized.

In this paper, Section II introduces the nondeterministic and deterministic entity-relationship relations as an enterprise-user's view of data. Both nondeterministic and deterministic relations are shown to have the same strength to characterize information about the real world. Section III describes a user schema in terms of regular or weak entity relations, or regular or weak relationship relations. An improved table form for the relations is introduced. Section IV concludes that both deterministic and nondeterministic relations for the user schema have the same strength to characterize information concerning entities and relationships which exist in our minds. Section V describes an entity-relationship approach to the logical database design. It can be used to construct the rela-

Manuscript received July 31, 1979; revised August 22, 1980.

The author is with the Department of Computer Sciences, University of Missouri, Columbia, MO 65211.

0098-5589/81/0100-0085$00.75

tions as a user's view of data in the entity-relationship model such that their relations are equivalent to the 3NF relations in the relational model. Section VI describes a method of representing physically the entity-relationship diagram or its corresponding user schema.

II. Entity-Relationship Relations

In this section, we shall define formally the nondeterministic and deterministic entity-relationship models of a database as an enterprise-user's view of that database in terms of a collection of time-varying relations of assorted degrees. We shall show that both nondeterministic and deterministic entity-relationship models have the same strength to characterize the information concerning entities and relationships.

Definition 1: Given a collection of sets $A_1, A_2, \cdots, A_n$ (not necessarily distinct), R is a *relation* on these n sets if it is a set of ordered n-tuples $<a_1, a_2, \cdots, a_n>$ such that $a_1 \in A_1$, $a_2 \in A_2, \cdots, a_n \in A_n$. Sets $A_1, A_2, \cdots, A_n$ are the *domains* of R. The value n is the *degree* of R.

Definition 2: Let e denote an entity, which is an object that can be distinctly identified. *An entity set* E is defined as $E = \{e \mid p(e)\}$, where p is the aforementioned test predicate.

Definition 3: Let $E = \{E_i \mid 1 \leqslant i \leqslant n\}$ be a collection of entity sets. *A relationship set* R *over* E is defined as $R = \{<e_1, e_2, \cdots, e_n> \mid e_i \in E_i, 1 \leqslant i \leqslant n\}$. Each tuple of entities $<e_1, e_2, \cdots, e_n>$ is a *relationship*.

The *role* of an entity in a relationship is the function that it performs in the relationship. The ordering of entities in the definition of relationship can be dropped if the roles of entities in the relationship are explicitly stated as follows: $<r_1/e_1, r_2/e_2, \cdots, r_n/e_n>$, where r_i is the role of e_i in the relationship.

Example 1: Let e denote an entity which exists in our minds. Entities can be classified into different entity types; each entity type contains a set of entities, each satisfying a set of predefined common properties. The set of entities is called an entity set, such as EMPLOYEE, PROJECT, PARENTS, and so forth. Because of different predefined properties, the entity sets are of different types. Thus, we call an entity set, an entity type.

In general, there will be associations or relationships linking the basic entities together. For example, as shown in Fig. 1, a MARRIAGE is a relationship between two entities in the entity set PARENTS. Their roles are HUSBAND and WIFE. We shall call MARRIAGE a relationship set, or a relationship type. PROJ-EMP and PROJ-MANAGER are two different types of relationship sets over two entity sets PROJECT and EMPLOYEE.

Fig. 1 is an extension of the entity-relationship diagrammatic notation in [2]. The entity sets PARENTS and CHILDREN are shown as double rectangular boxes, each of which is connected by a directed edge from the entity set EMPLOYEE and the relationship set E (existence dependency) and EMP-CHILD, respectively. The diagram expresses the existence and identification dependencies of the entity set PARENTS on EMPLOYEE; it indicates that the existence of any entity in the entity set PARENTS depends on the corresponding entity in the entity set EMPLOYEE; that is, if an employee leaves the company, his parents may no longer be of interest. It also indicates that parents are identified by their own names and by the values of the primary key of the employees supporting them.

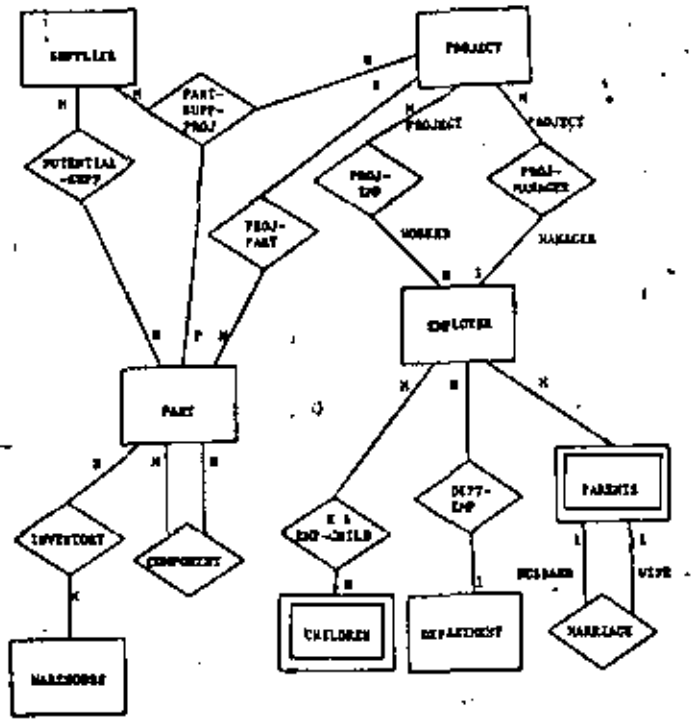


Fig. 1. An entity-relationship diagram for an enterprise schema.

If a relationship set E, represented in a diamond-shaped box, is introduced between the entity sets EMPLOYEE and CHILDREN, then it expresses the existence dependency of the entity set CHILDREN on EMPLOYEE, and the identification dependency of the entity set CHILDREN on EMPLOYEE does not exist. If the same diamond-shaped box has the names E and EMP-CHILD, then the relationship set EMP-CHILD of the entity sets EMPLOYEE and CHILDREN includes the existence dependency of the entity set CHILDREN on EMPLOYEE. If the same diamond-shaped box has the name EMP-CHILD, then the relationship set EMP-CHILD includes the existence and identification dependencies of the entity set CHILDREN on EMPLOYEE. Without this diamond-shaped box, it indicates the existence and identification dependencies of the entity set CHILDREN on EMPLOYEE.

Definition 4: A *value set* V is defined as $V = \{v \mid p(v)\}$, where p is the aforementioned test predicate.

Definition 5: Let $E = \{E_i \mid 1 \leqslant i \leqslant n\}$ be a collection of finite entity sets. Let $V = \prod_{j=1}^{m} V_j$, $1 \leqslant m$, be a Cartesian product of value sets. Let $R \subseteq \prod_{j=1}^{n} E_j$, $1 \leqslant n$. A *multivalued attribute* F is defined as $F \subseteq R \times V$. If $m = 1$, then V is a value set. If $n = 1$, then R is an entity set, and F is a mapping from an entity set into a value set or a Cartesian product of value sets. If $n > 1$, then R is a relationship set over E, and F is a mapping from a relationship set into a value set or a Cartesian product of value sets. In particular, if F is a function which maps R into V, in notation $F: R \to V$, then F is called a *single-valued attribute*.

Example 2: Consider the entity sets PROJECT and EMPLOYEE and their relationship set PROJ-EMP, shown in Fig. 2. The attribute STARTING-DATE is the date that an employee starts working for a particular project, and the attribute %-OF-TIME is the percentage of time that an employee is expected to spend on a particular project. The concept of attributes of a relationship is important in understanding the semantics of

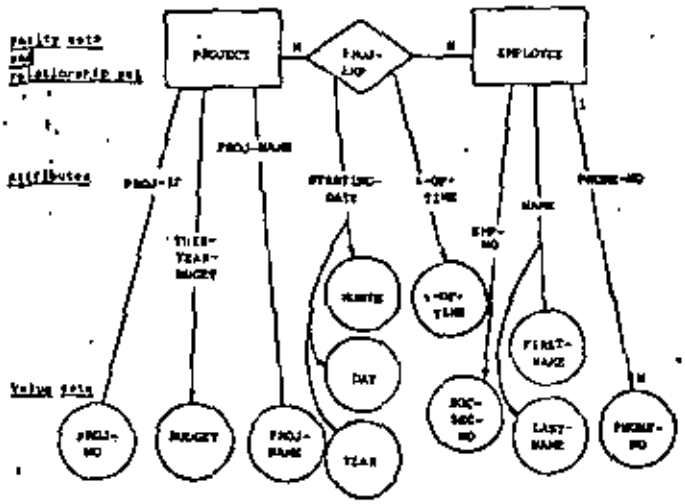


Fig. 2. Value sets of different types and attributes of entity sets and relationships.

data. This concept is similar to the relationship data in network (CODASYL) type database systems, and similar to the intersection data in hierarchical type (IMS type) database systems.

In some cases, an attribute may have more than one value for a given entity. For instance, the PHONE-NO of an employee x may have more than one value. In this case, we put $1:N$ beside the arrow to indicate that it is a multivalued attribute. If the home phone numbers and the office phone numbers of the employees are of interest, then HOME-PHONE-NO × OFFICE-PHONE-NO, which is a repeating group, may be taken into consideration. Both vector (the Cartesian product of value sets) and repeating group are data-aggregates of different types. For the latter, both HOME-PHONE-NO and OFFICE-PHONE-NO are referred to be individually addressable groups of data items. Thus, in the entity-relationship diagram, both HOME-PHONE-NO and OFFICE-PHONE-NO should be created. Each of these attributes maps the entity set EMPLOYEE into a value set PHONE-NO.

Definition 6: An entity-relationship diagram (ERD) is a diagram which consists of a collection of entity sets and relationship sets and their associations, which are in the upper conceptual domains, and the attributes and value sets which are needed to describe the properties of some of the entities and relationships which may be of interest, in the lower conceptual domains.

Example 3: Fig. 1 illustrates the associations of the entity and relationship sets. Fig. 2 illustrates the attributes and value sets needed to describe the properties of some of the entities and relationships which may be of interest to an enterprise. A diagram of Figs. 1 and 2 is an ERD.

Definition 7: A nondeterministic entity-relationship relation M over E (NERR) is defined as $M = (E, R, V, F)$, where

$E = \{E_i | 1 \leqslant i \leqslant n\}$ is a collection of countable entity sets,

$R \subseteq \prod_{i=1}^{n} E_i$ is a predefined relationship set over E;

if $n = 1$, then $R = \phi$,

$V = \{V_i | 1 \leqslant i \leqslant m\}$ is a collection of countable value sets,

and

$$F = \left\{ F_i | F_i \subseteq (E_\alpha \cup R) \times \prod_{\alpha}^{\beta_i} V_{i_j}, 1 \leqslant v_i \leqslant p \right\},$$

is a finite set of multivalued attributes.

Let E_i be a finite set of distinct entities. Let E_i' and E_i'' be subsets of E_i. If $R \subseteq E_i' \times E_i''$, then we can always write E_i into entity sets E_i' and E_i''. For this reason, without loss of generality, we can impose a restriction that if E is a singleton set, then $R = \phi$; that is, no relationship set is to be defined on the singleton set E.

We may restrict F into a set of functions F_i. Each F_i is a function which maps an entity set or a predefined relationship set R into a value set or a Cartesian product of value sets.

Definition 8: In M, if $F = \{F_i | F_i: (E_\alpha \cup R) \to \Pi_{j_i}^{\alpha} V_{i_j}, 1 \leqslant v_i \leqslant p\}$ is a finite set of attributes, then M is said to be *a deterministic entity-relationship relation M over E* (DERR).

Definition 9: A deterministic entity-relationship model (DERM) is defined as a collection of deterministic entity-relationship relations; otherwise, it is *a nondeterministic entity-relationship model* (NERM).

In the sequel, we shall use the term, entity-relationship relation, if the nondeterminism and determinism are not needed to clarify.

It also should be noted that the entity-relationship relation $M = (E, R, V, F)$ can be displayed in a table form, shown in Fig. 3, which demonstrates information about entities in an entity set or information about relationships in a relationship set. Each row of values is related to a relationship, which is indicated by a group of entities, each having a specific role and belonging to a specific entity set. Each column is related to a value set which, in turn, is related to an attribute. The ordering of rows (tuples) and columns is insignificant. The whole table shown in Fig. 3 is a deterministic entity-relationship relation, if each distinct relationship has one and only one value tuple.

In Fig. 3, that both the relation names and the relation scheme in the table are used for describing the structure of the relation $M = (E, R, V, F)$ is called its intention, which is static. That the relations in the table are used for denoting a set of tuples having the appropriate structure, is called its extension. In this paper, we shall use the term relation scheme to refer to an intention, that is to refer to a structural description of an entity-relationship relation, and the term relational schema to refer to a collection of intentions.

Example 4: The entity-relationship diagram shown in Fig. 2, can be organized in a collection of table forms, shown in Figs. 4-6. They contain information about entities in the entity sets, EMPLOYEE and PROJECT, and information about relationships between two entity sets, EMPLOYEE and PROJECT, in a relationship set PROJ-EMP. In Fig. 6, the attributes STARTING-DATE and %-OF-TIME are the attributes which map the relationships of the relationship set PROJ-EMP into the value sets MONTH × DAY × YEAR and %-OF-TIME, respectively. They are neither the attributes of EMPLOYEE nor the attributes of PROJECT, since their meanings depend on both the employee and projects involved.

In the remaining section, we shall show that both NERM and

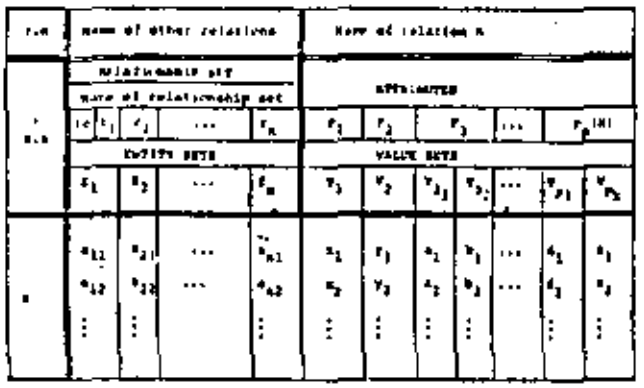


Fig. 3. A table of an entity-relationship relation M. $r.n$, $r.s$, r, ro, and $F(N)$ denote relation names, relation scheme, relations, roles, and multivalued attribute F, respectively.

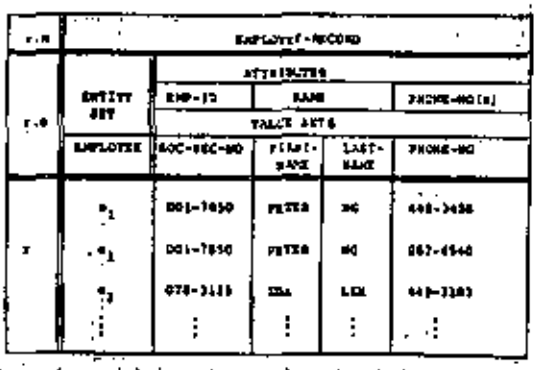


Fig. 4. A nondeterministic entity-relationship relation, EMPLOYEE-RECORD: information about entities in an entity set, EMPLOYEE.

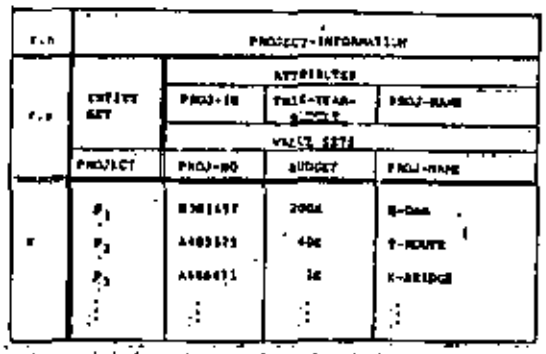


Fig. 5. A deterministic entity-relationship relation PROJECT-INFORMATION.

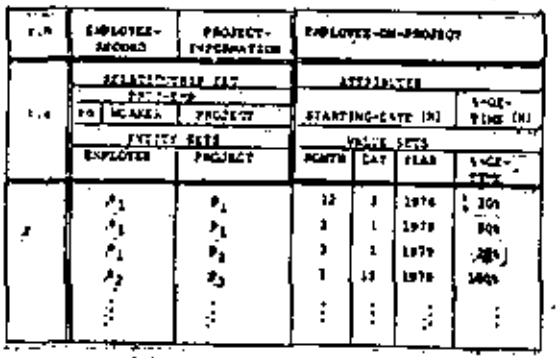


Fig. 6. A nondeterministic entity-relationship relation, EMPLOYEE-ON-PROJECT: information about relationships in the relationship set PROJ-EMP.

DERM have the same strength to characterize the information about entities of a given entity set or the information about relationships of a given relationship set. Given any of these entities and relationships, the associated information can be retrieved in an exact form.

Definition 10: Let E be an entity set. Let $R \subseteq \Pi_{i=1}^{n} E_i$ be a relationship set over a collection of entity sets E_i, $1 \leqslant i \leqslant n$. Let $F = \{F_i | F_i \subseteq (E \cup R) \times \Pi_{j=1}^{n_i} V_{ij},\ 1 \leqslant i \leqslant m,\ 1 \leqslant n_i\}$ be a set of attributes. Let $K = \{F_1, F_2, \cdots, F_n | 1 \leqslant n\}$ be a subset of F.

Then, for any e in E, *the operator K of an entity e* is defined as

$$\begin{aligned} K(e) &= \{F_1, F_2, \cdots, F_n\}(e) \\ &= (F_1(e), F_2(e), \cdots, F_n(e)) \\ &= \left\{(x_1, x_2, \cdots, x_n) | x_i \in F_i(e) \text{ and } x_i \in \prod_{j=1}^{n_i} V_{ij}, 1 \leqslant i \leqslant n\right\}. \end{aligned}$$

Let r be a relationship $\langle e_1, e_2, \cdots, e_n \rangle$ in R. *The operator K of a relationship r* is defined as

$$\begin{aligned} K(r) &= \{F_1, F_2, \cdots, F_n\}(r) \\ &= (F_1(r), F_2(r), \cdots, F_n(r)) \\ &= \left\{(x_1, x_2, \cdots, x_n) | x_i \in F_i(r) \text{ and } x_i \in \prod_{j=1}^{n_i} V_{ij}, 1 \leqslant i \leqslant n\right\}. \end{aligned}$$

Definition 11: Let $M = (E, R, V, F)$ be an entity-relationship relation. Let x be a tuple $(x_1, x_2, \cdots, x_n)$. We say that the tuple x, an information of an entity e in the entity set E (or an information of a relationship r in the relationship set R), can be *derived* from M if $F = \{F_1, F_2, \cdots, F_n\}$ is the set of attributes of M such that $x \in F(e)$ (or $x \in F(r)$) is in M.

Definition 12: Consider two entity-relationship relations $M = (E, R, V, F)$ and $M' = (E', R', V', F')$, where $E = \{E_i | 1 \leqslant i \leqslant n\}$ and $E' = \{E'_i | 1 \leqslant i \leqslant m\}$. Let $S \subseteq E \cap E'$. (If R and R' are not empty, then let $R \subseteq S \times \Pi_{i=1}^{q} E_i$, where $1 \leqslant q \leqslant n$, $E_i \in E$, and $E_i \notin S$, and let $R' \subseteq S \times \Pi_{i=1}^{p} E'_i$, where $1 \leqslant p \leqslant m$, $E'_i \in E'$, and $E'_i \notin S$.)

We say that M *preserves M' under S* if and only if any information $(x_1, x_2, \cdots, x_k)$ of s in S, that can be derived from M', can also be derived from M, where $1 \leqslant k \leqslant |F|$, $1 \leqslant k \leqslant |F'|$, and $x_i \in \Pi_{j=1}^{n_i} V_{ij}$ and $x_i \in \Pi_{j=1}^{n'_i} V'_{ij}$, $V_{ij} \in V$, and $V'_{ij} \in V'$.

We say that M *strongly preserves M' under S* if and only if M preserves M' under S and M' also preserves M under S.

Let $D = \{M | M = (E, R, V, F)$ is an entity-relationship relation$\}$ and $D' = \{M' | M' = (E', R', V', F')$ is an entity-relationship relation$\}$ be the entity-relationship models. (Let $S = \{S'|$ for every $M = (E, \phi, V, F)$ in D and $M' = (E', \phi, V', F')$ in D', $S' \subseteq E \cap E'$, or for every $M = (E, \Pi_{i=1}^{p} E_i, V, F)$ and $M' = (E', \Pi_{i=1}^{q} E'_i, V', F')$, $S' \subseteq \{E_i | 1 \leqslant i \leqslant p\} \cap \{E'_i | 1 \leqslant i \leqslant q\}\}$. We say that *the model D preserves the model D' under S* if and only if, for every entity-relationship relation M' in D', there corresponds some entity-relationship relation M in D such that M preserves M' under S'.

We also say that *the model D strongly preserves the model D' under S* if and only if D preserves D' under S, and D' preserves D under S.

It should be noted that, in models D and D', S is a collection

of subsets of entity sets of relations in D and D'; or a collection of subsets of involved entity sets in the relationship sets in D and D'.

Theorem 1: Let $M = (E, R, V, F)$ be a NERR. There corresponds a DERR $M' = (E', R', V, F')$ such that M' strongly preserves M under $E \cup R$. The converse also holds.

Proof: For every $F_i \subseteq R \times \prod_{j=1}^{v_i} V_{ij}$ in M, create arbitrarily an entity set E_i' such that E_i' and $\prod_{j=1}^{v_i} V_{ij}$ is one-to-one corresponding. Then, let $R' \subseteq R \times \prod_{j=1}^{v_i} V_{ij}$. Construct $F' \subseteq R' \times \prod_{j=1}^{v_i} V_{ij}$ as follows. Whenever $r \in R$, $F(r) = \{F_1, F_2, \cdots, F_m\}(r) = \{(x_1, x_2, \cdots, x_m) | x_i \in \prod_{j=1}^{v_i} V_{ij}, 1 \leqslant i \leqslant m\}$ is in M, then for each $(x_1, x_2, \cdots, x_m) \in F(r)$, we have $<r, x_1, x_2, \cdots, x_m> \in R'$, $F'(<r, x_1, x_2, \cdots, x_{i-1}, x_i, x_{i+1}, \cdots, x_m>) = x_i$ is defined by $F_i' \subseteq R' \times \prod_{j=1}^{v_i} V_{ij}$ for $1 \leqslant i \leqslant m$, and $F'(<r, x_1, x_2, \cdots, x_m>) = \{F_1', F_2', \cdots, F_m'\}(<r, x_1, x_2, \cdots, x_m>) = (x_1, x_2, \cdots, x_m)$ is in M'. Clearly, F_i': $R' \to \prod_{j=1}^{v_i} V_{ij}$ and M' strongly preserves M under R. From the definitions, the converse also holds.

Corollary 1: The NERM and DERM have the same strength to characterize the same information about entities of a collection of entity sets or the same information about relationships of a collection of relationship sets.

Proof: It follows from the definitions and Theorem 1.

In the entity-relationship approach to the logical database design, one of the major steps [2] is to identify the properties of entities and relationships, which are of interest to the enterprise. That is, we wish to identify attributes and value types for the entities and relationships. In order to reduce the amount of redundancy in the stored data and to minimize the problem of inconsistency in the stored data, the functional attributes are desired. However, in the process of logical database design, it becomes difficult to design a database if we are allowed to use only functional attributes. By Theorem 1, we can use relational attributes and value types to identify the properties of entities and relationships, and then we transform these relational attributes into functional attributes.

III. Entity and Relationship Relations

In this section, we shall identify an entity-relationship relation into one of the four types: a regular or weak entity relation, or a regular or weak relationship relation. They are different from the entity-relationship relation in the sense that they identify the entities or relationships of the entity-relationship relation by values, which are elements of a Cartesian product of value sets or elements of a value set.

In Section II, we have defined the operator K of an entity e and the operator K of a relationship r. We extend the operator K into a set operator $\{K_1, K_2, \cdots, K_t\}$; each K_i contains a set of attributes F_{i_m}.

Definition 13: Let E be an entity set. Let $E_i = \{E_{ij} | 1 \leqslant j \leqslant n_i\}$ be a collection of entity sets, where $1 \leqslant i \leqslant t$. Let $R \subseteq \prod_{j=1}^{n_i} E_{ij}$ be a relationship set. Let $R \subseteq \prod_{j=1}^{t} \{E \cup R_j\}$. Let $F_i = \{F_{i_m} | F_{i_m} \subseteq \{E \cup R_i\} \times \prod_{j=1}^{v_{i_m}} V_{i_m j}\}$, $1 \leqslant i \leqslant t$. Let $K_i = \{F_{i_1}, F_{i_2}, \cdots, F_{i_m}\}$, $1 \leqslant i \leqslant t$. Let $K = \{K_1, K_2, \cdots, K_t\}$.

Then, for any $<r_1, r_2, \cdots, r_t> \in R$, where $r_i \in E \cup R_i$, *the operator K of a relationship* $<r_1, r_2, \cdots, r_t>$ is defined recursively as follows:

$$K_i(r_i) = \{F_{i_1}, F_{i_2}, \cdots, F_{i_m}\}(r_i), \quad \text{for every } 1 \leqslant i \leqslant t, \tag{1}$$

$$K_i(r_i) \text{ is undefined for } 1 \leqslant i \neq j \leqslant t, \tag{2}$$

and

$$\begin{aligned} K(<r_1, r_2, \cdots, r_t>) &= \{K_1, K_2, \cdots, K_t\} \\ &\quad \cdot (<r_1, r_2, \cdots, r_t>) \\ &= (K_1(<r_1, r_2, \cdots, r_t>), \\ &\quad K_2(<r_1, r_2, \cdots, r_t>), \cdots, \\ &\quad K_t(<r_1, r_2, \cdots, r_t>)) \\ &= (K_1(r_1), K_2(r_2), \cdots, K_t(r_t)). \end{aligned} \tag{3}$$

Definition 14: Let $M = (E, R, V, F)$ be an entity-relationship relation, where $E = \{E_i | 1 \leqslant i \leqslant n\}$; $R \subseteq \prod_{i=1}^{n} E_i$, $1 < n$, and if $n = 1$, the $R = \phi$, and $F = \{F_i | F_i \subseteq \{E_\alpha \cup R\} \times \prod_{j=1}^{v_i} V_{ij}, 1 \leqslant v_i \leqslant p\}$. Then K, a subset of F, is a *superkey* of M if, for every r and r' in $E_\alpha \cup R$, $r \neq r'$ if and only if $K(r) \neq K(r')$, and both $K(r)$ and $K(r')$ are singleton sets.

In particular, if E is an entity set (i.e., R is empty), then $F = \{F_i | F_i \subseteq E \times \prod_{j=1}^{v_i} V_{ij}, 1 \leqslant v_i \leqslant p\}$. Then K, a subset of F, is a *superkey* of M if, for every e and e' in E, $e \neq e'$ if and only if $K(e) \neq K(e')$, and both $K(e)$ and $K(e')$ are singleton sets.

The set K is a *key* of M if it is a minimal superkey. K is the *primary key* of M if the minimal superkey K is selected as key of M.

Clearly, an entity-relationship relation M can have many keys. A superkey of M is any set of attributes in M that contains a key of M. Every key is also a superkey. Furthermore, if M is a deterministic entity-relationship relation, then K, a subset of F, is a superkey of M if, for every r and r' in $E_\alpha \cup R$, $r \neq r'$ if and only if $K(r) \neq K(r')$, and both $K(r)$ and $K(r')$ are singleton sets.

Based on the methods of identifying entities of relationships, the entity-relationship relations can be identified into four types. Without loss of generality, let K and F be disjoint sets of attributes.

Definition 15: Let $E = \{e_i | 1 \leqslant i \leqslant \alpha\}$ be a countable entity set. *A nondeterministic regular entity relation* (NRER) M *over* E is defined as $M_E = (K, R, V, F)$ where $R = \phi$, $V = \{V_i | 1 \leqslant i \leqslant m\}$ is a collection of value sets, $K = \{F_1', F_2', \cdots, F_n' | F_i' \subseteq E \times \prod_{j=1}^{v_i} V_{ij}, F_i'$ is not in $F, 1 \leqslant i \leqslant n\}$ is the primary key, and $F = \{F_i |$ for every $1 \leqslant i \leqslant k, F_i \subseteq K(E) \times \prod_{j=1}^{v_i} V_{ij}, 1 \leqslant v_i\}$ is the set of the attributes.

Definition 16: In M_E, if $F = \{F_i |$ for every $1 \leqslant i \leqslant k$, F_i: $K(E) \to \prod_{j=1}^{v_i} V_{ij}, 1 \leqslant v_i\}$ is the set of (functional) attributes, then M_E is said to be *a deterministic regular entity relation* (DRER) M *over* E.

Definition 17: Let $E' = \{E_i | 1 \leqslant i \leqslant n\}$ be a collection of countable entity sets. Let $E \notin E'$ be a countable entity set. Let K_i be the primary keys of the involved entities E_i. *A nondeterministic weak entity relation* (NWER) M *over* E *and* E' is defined as $M_{E,E'} = (K, R, V, F)$ where $R \subseteq \prod_{i=1}^{n} E_i \times E$ is a relationship set over E' and E, $V = \{V_i | 1 \leqslant i \leqslant m\}$ is a collection of value sets, $K = \{K_1, K_2, \cdots, K_n, F_1', F_2', \cdots, F_k' | 1 \leqslant i \leqslant n, 1 \leqslant \alpha \leqslant k, K_i \subseteq E_i \times \prod_{j=1}^{v_i} V_{ij}$, where $1 \leqslant v_i$ and

$V_{i_j} = \Pi_{s=1}^{p} V'_s$, $1 \leq p$, for some $V'_s \in V$, and $F'_\alpha \subseteq F \leq \Pi_{\beta=1}^{\alpha} V_{\alpha_j}$, $V_{\alpha_j} \in V\}$ is the primary key, and $F = \{F_i | \text{ for } 1 \leq i \leq q, F_i \subseteq K(R) \times \Pi_{j=1}^{v_i} V_{i_j}, 1 \leq v_i\}$ is the set of attributes.

Definition 18: In $M_{E,E'}$, if $F = \{F_i | \text{ for } 1 \leq i \leq q, F_i: K(R) \to \Pi_{j=1}^{v_i} V_{i_j}, 1 \leq v_i\}$ is the set of attributes, then $M_{E,E'}$ is said to be *a deterministic weak entity relation* (DWER) *M over E and E'*.

Definition 19: Let $E = \{E_i | 1 \leq i \leq n\}$ be a collection of countable entity sets. Let K_i be the primary keys of the involved entities in the entity set E_i. Then *a nondeterministic regular relationship relation* (NRRR) *M over E* is defined as $M_E = (K, R, V, F)$ where $R \subseteq \Pi_{j=1}^{n} E_j$ is a relationship set over E, $V = \{V_i | 1 \leq i \leq v\}$ is a set of value sets, $K = \{K_1, K_2, \cdots, K_n | 1 < n, 1 \leq i \leq n, K_i \subseteq E_i \times \Pi_{j=1}^{v_i} V_{i_j}$, where $1 \leq v_i$ and $V_{i_j} = \Pi_{s=1}^{p} V'_s$, $1 \leq p$ and for some $V'_s \in V\}$ is the primary key, and $F = \{F_i | \text{ for } 1 \leq i \leq k, F_i \subseteq K(R) \times \Pi_{j=1}^{v_i} V_{i_j}, 1 \leq v_i\}$ is the set of attributes.

Definition 20: In M_E, if $F = \{F_i | \text{ for } 1 \leq i \leq k, F_i: K(R) \to \Pi_{j=1}^{v_i} V_{i_j}, 1 \leq v_i\}$ is the set of attributes, then M_E is said to be *a deterministic regular relationship relation* (DRRR) *M over E*.

Definition 21: Let $E = \{E_i | 1 \leq i \leq n\}$ and $E' = \{E'_i | 1 \leq i \leq m\}$ be disjoint collections of entity sets. Let K'_i be the primary keys of the involved entities in E'_i. Let K_i be the primary keys of the involved entities in E_i. Then *a nondeterministic weak relationship relation* (NWRR) *M over E and E'* is defined as $M_{E,E'} = (K, R, V, F)$, where $R \subseteq \Pi_{j=1}^{m} E'_j \times \Pi_{j=1}^{n} E_j$ is a relationship set over E and E', $V = \{V_i | 1 \leq i \leq v\}$ is a collection of value sets, $K = \{K'_1, K'_2, \cdots, K'_m, K_1, K_2, \cdots, K_n | 1 \leq m, 1 < n$, for $1 \leq \alpha \leq m, K'_\alpha \subseteq E'_\alpha \times \Pi_{\beta=1}^{\alpha} V_{\alpha_j}$, where $1 \leq v_\alpha$ and $V_{\alpha_j} = \Pi_{q=1}^{p} V'_q$, $1 \leq p$ and $V'_q \in V$, and for $1 \leq s \leq n, K_s \subseteq E_s \times \Pi_{j=1}^{v_s} V_{s_j}$, where $1 \leq v_s$ and $V_{s_j} = \Pi_{q=1}^{p} V''_q$, $1 \leq p$ and $V''_q \in V\}$ is the primary key, and $F = \{F_i | \text{ for } 1 \leq i \leq k, F_i \subseteq K(R) \times \Pi_{j=1}^{v_i} V_{i_j}, 1 \leq v_i\}$ is the set of attributes.

Definition 22: In $M_{E,E'}$, if $F = \{F_i | \text{ for } 1 \leq i \leq k, F_i: K(R) \to \Pi_{j=1}^{v_i} V_{i_j}, 1 \leq v_i\}$ is the set of attributes, then $M_{E,E'}$ is said to be *a deterministic weak relationship relation* (DWRR) *M over E and E'*.

Example 5: Figs. 7–13 demonstrate the table forms of the regular entity, weak entity, regular relationship, and weak relationship relations. This design of table forms does provide to the users a clarity, simplicity, and unity of the concepts and the structure of the entity-relationship relations.

In Fig. 9, the involved entities in the regular relationship relation are represented by their primary keys, EMP-ID and PROJ-ID of their regular entity relations, EMPLOYEE-RECORD, and PROJECT-INFORMATION, respectively. The role names provide the semantic meaning for the values in the corresponding columns. In the regular entity relation CHILDREN-RECORD, shown in Fig. 11, the dependents are identified by their children identification, CHILD-ID; NAME and AGE are attributes of the dependents. But, the appearance of EMPLOYEE-RECORD affiliated with the CHILDREN-RECORD indicates that the existence of the entity set CHILDREN depends on EMPLOYEE. However, since this existence relationship of the entity sets EMPLOYEE and CHILDREN without any identification purpose, it must be represented in a table form given in Fig. 10, and does not have any attribute and value. In the relation

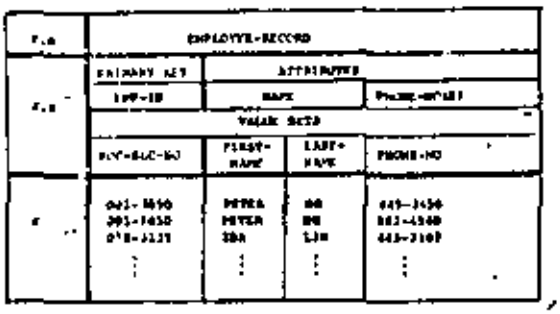


Fig. 7. A nondeterministic regular entity relation EMPLOYEE-RECORD.

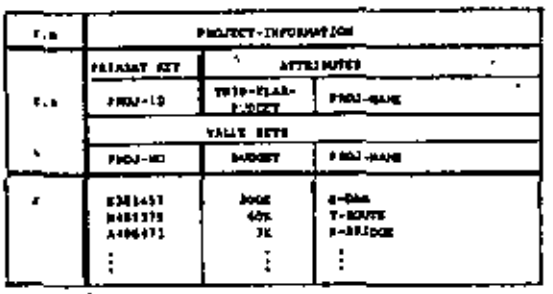


Fig. 8. A deterministic regular entity relation PROJECT-INFORMATION.

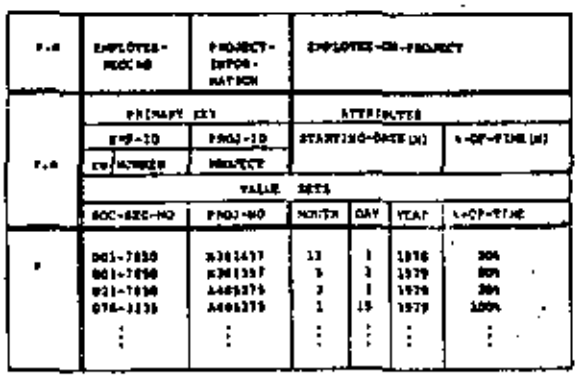


Fig. 9. A nondeterministic regular relationship relation, EMPLOYED-ON-PROJECT.

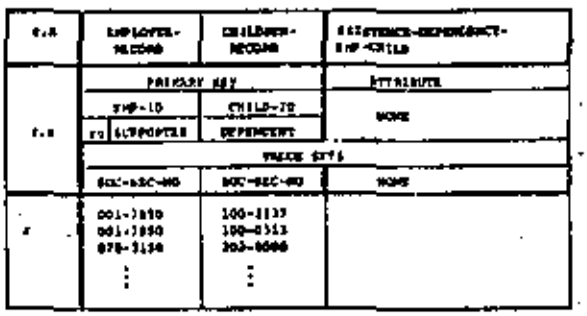


Fig. 10. A deterministic regular relationship relation, EXISTENCE-DEPENDENCY-EMP-CHILD.

PARENTS-RECORD, shown in Fig. 12, the parents are identified by their names and by the values of the primary key of the employees supporting them. Therefore, the appearance of the EMPLOYEE-RECORD affiliated with the PARENTS-RECORD indicates the existence and identification dependencies of the entity set PARENTS on another entity set EMPLOYEE. PHONE-NO is the only attribute of the parents. Fig. 13 is a weak relationship relation, in which, YEAR-OF-MARRIAGE is an attribute of the relationship set MARRIAGE over the entity set PARENTS. Each of the relationships is identified by the NAME of FATHER and NAME of MOTHER and their children's EMP-ID supporting

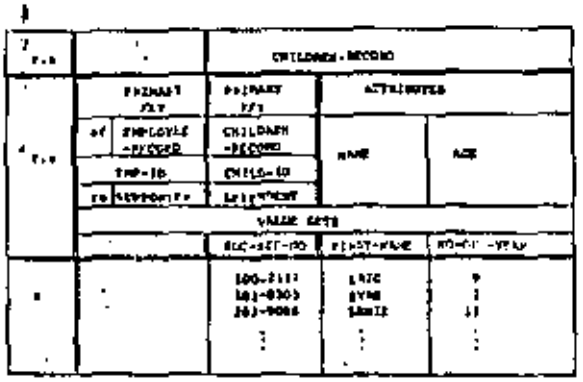


Fig. 11. A deterministic regular entity relation, CHILDREN-RECORD. The table also shows the existence dependency of CHILDREN on EMPLOYEE.

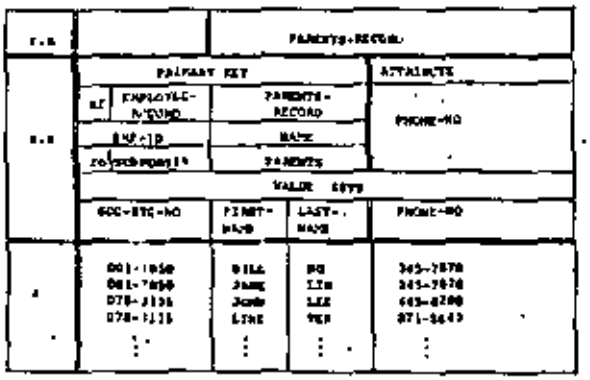


Fig. 12. A deterministic weak entity relation, PARENTS-RECORD. The table also shows the existence and identification dependencies of PARENTS on EMPLOYEE.

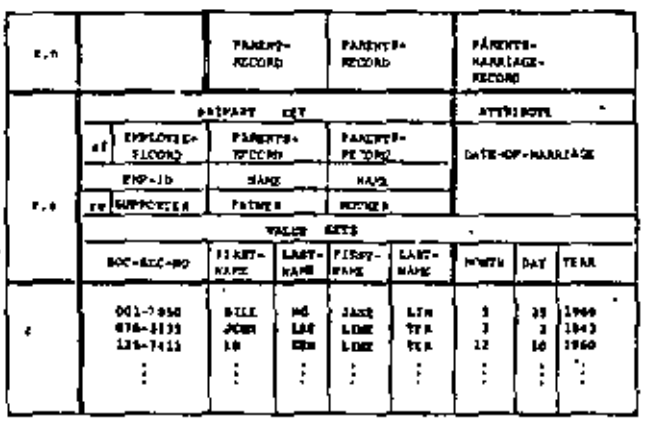


Fig. 13. A deterministic weak relationship relation, PARENTS-MARRIAGE-RECORD. The table also shows the existence and identification dependencies of PARENTS on EMPLOYEE.

them. The appearance of EMPLOYEE-RECORD affiliated with the PARENTS-RECORD also indicates the existence and identification dependencies of the entity set PARENTS on another entity set EMPLOYEE.

In conclusion, there corresponds a relation (of table form) for every entity set or relationship set in the entity-relationship diagram. For any entity set E', the identification of the entities in E' is dependent on another entity set E if and only if the relation for the entity set E' is a weak entity relation in which the primary key is the combination of the primary key in the relation for E and an attribute (or possibly a set of attributes) in the relation for E'. Furthermore, if there is a relationship set R over the entity set E' and others, then the relation for the relationship set R is a weak relationship relation, and vice versa. Otherwise, the relation for any entity set or any relationship set is a regular entity relation or a regular relationship relation, respectively.

Definition 23: *A deterministic entity-relationship model* D is defined as $D = \{M\}$ for every M, M is one of the deterministic types: a regular or weak entity relation, or a regular or weak relationship relation}. In D, if one of the relations is nondeterministic, then D is said to be *a nondeterministic entity-relationship model.*

Definition 24: Let $M = (K, R, V, F)$ be one of the regular or weak relation, or regular or weak relationship relation. Let x be a tuple $(x_1, x_2, \cdots, x_n)$. We say that the tuple x, an information of a given k, which is a value of the primary key K, can be *derived* from M if $F = \{F_1, F_2, \cdots, F_n\}$ is the set of attributes of M such that $x \in F(k)$ is in M.

Note that k is either a value in V_i, for some V_i in V, or a tuple in $\prod_{i=1}^{n} V_i$, for V_i in V, and k represents uniquely an entity or a relationship, respectively.

Definition 25: Consider two relations: $M = (K, R, V, F)$ and $M' = (K', R', V', F')$ which are one of the types: a regular or weak entity relation, or a regular or weak relationship relation. Let $S \subseteq K \cap K'$. We say that M *preserves* M' *under* S if and only if any information $(x_1, x_2, \cdots, x_k)$ of s, which is a value of S, that can be derived from M', can also be derived from M, where $1 \leqslant k \leqslant |F|$ and $1 \leqslant k \leqslant |F'|$, and $x_i \in \prod_{j=1}^{n_i} V_{ij}$ and $x_i \in \prod_{j=1}^{n'_i} V'_{ij}$, V_{ij} in V and V'_{ij} in V'. We say that M *strongly preserves* M' *under* S if and only if M preserves M' under S, and M' preserves M under S.

Similarly, we say that the model D preserves the model D' under $S = \{S' | S' \subseteq K \cap K'\}$, if and only if for every relation $M' = (K', R', V', F')$ in D', there corresponds some relation $M = (K, R, V, F)$ in D such that M preserves M' under S' in S.

We also say that *the model D strongly preserves the model* D' *under* S if and only if D preserves D' under S and D' preserves D under S.

It should be noted that, in the relations M and M', S is a subset of K, where K is described by a group of attributes that are used to identify uniquely entities in the entity set, or relationships in the relationship set. s is an attribute value, which is an element of a Cartesian product of value sets or simply a value set. For the model D and D', S is a collection of attribute sets S'; each attribute set S' is a subset of K' and K.

IV. Relationship Between Entity (Relationship) Relation and Entity-Relationship Relation

In this section, we shall show that the entity-relationship relations can be translated exactly into relations of the user schema. We shall demonstrate the fact that each tuple of any entity-relationship relation can be uniquely identified by some primary keys. Basically, any entity key is a group of attributes such that the mapping from the entity set onto the corresponding group of value sets is one-to-one. If we cannot find such a one-to-one mapping on available data, or if simplicity in identifying entities is desired, we may define an artificial attribute and a value set so that such mapping is possibly defined. In the case where several keys exist, we usually choose a semantically meaningful key as the entity primary key.

In certain cases, the entities in an entity set cannot be uniquely identified by the values of their own attributes; then we must use a relationship(s) to identify them. Theoretically, many kinds of relationships may be used to identify entities. The method of identification of entities by relationships with other entities can be applied recursively until the entities which can be identified by their own attributes' values are reached. Therefore, we have two forms of entity relations. If relationships are used for identifying the entities, we shall call it a weak entity relation. If relationships are not used for identifying the entities, we shall call it a regular entity relation.

Since a relationship is identified by the involved entities, the primary key of a relationship can be represented by the primary keys of the involved entities. We also have two forms of relationship relations. If all entities in the relationship are identified by their own attributes' values, we shall call it a regular relationship relation. If some entities in the relationship are identified by other relationships, we shall call it a weak relationship relation. Thus, we state the following.

Theorem 2: For any entity-relationship relation $M = (E, R, V, F)$, there corresponds a regular or weak entity relation, or a regular or weak relationship relation $M' = (K, R', V, F)$ such that M' strongly preserves M under $K(E \cup R)$, the identification of entities or relationships. The converse is true.

Proof: Let $M_{E,E'} = (K, R, V, F)$ be a weak relationship relation over E and E'. Then, we can construct an entity-relationship relation $M' = (E'', R', V, F')$ as follows. Let t_1 be a tuple $<e_1', e_2', \cdots, e_m', e_1, e_2, \cdots, e_n>$ in $E' \times E$. Let t_2 be a tuple $(z_1, z_2, \cdots, z_{m1}, x_1, x_2, \cdots, x_n)$ in $\prod_{i=1}^{m} \prod_{j=1}^{v_i} V_{ij} \times \prod_{i=1}^{n} \prod_{j=1}^{v_{z_i}} V_{z_{ij}}$, where V_{ij} and $V_{z_{ij}}$ are in V. For each $K(t_1) = t_2$, we have $E'' = \{\overline{E}_i' | 1 \leq i \leq n\} \cup \{\overline{E}_j' | 1 \leq j \leq m\}$ such that $z_j \in \overline{E}_j'$, for $1 \leq j \leq m$ and $x_i \in \overline{E}_i$, for $1 \leq i \leq n$, and $t_1 \in R' \subseteq \prod_{i=1}^{m} \overline{E}_i' \times \prod_{i=1}^{n} \overline{E}_i$. $F' = \{P_i | P_i \subseteq R' \times \prod_{j=1}^{v_i} V_{ij}, 1 \leq i \leq k, 1 \leq v_i$ such that for every i, $F_i \in F$, $y_i \in F_i(K(t_1))$, where $K(t_1) = t_2$ is defined by $F_i \subseteq K(R) \times \prod_{j=1}^{v_i} V_{ij}$ if and only if $y_i \in P_i(t_2)$ is defined by $P_i\} \cup \{P_i' | 1 \leq i \leq n, P_i' \subseteq R' \times \prod_{j=1}^{v_i} V_{ij}, 1 \leq v_i$ such that $P_i'(t_2) = x_i \in \prod_{j=1}^{v_i} V_{ij}$ if and only if $x_i \in K_i(e_i)$ for $e_i \in E_i$ and $K(t_1) = t_2\}$. For $1 \leq i \leq n$, $K_i = \{F_{ij} | 1 \leq j \leq \alpha\}$, $P_i' = \{P_{ij}' | 1 \leq j \leq \alpha$, $P_{ij}' \subseteq R' \times \prod_{j=1}^{p} V_j'$, such that $P_{ij}'(t_2) = x_i = (x_{i_1}, x_{i_2}, \cdots, x_{ij-1}, x_{ij}, x_{ij+1}, \cdots, x_{i_\alpha}) \in \prod_{j=1}^{v_{ij}} V_{ij}$, where $V_{ij}' = \prod_{j=1}^{p} V_j'$ if and only if $x_i \in K(e_i) = (F_{i_1}, F_{i_2}, \cdots, F_{ij-1}, F_{ij}, F_{ij+1}, \cdots, F_{i_\alpha})(e_i)$, and $K(t_1) = t_2\}$. Clearly, M' is an entity-relationship relation such that M' strongly preserves M under $K(R)$. Similarly, it can be shown in the same manner that the other types of relations can be transformed into entity-relationship relations, without loss of information. And so is the converse.

Corollary 2: For any entity-relationship model D, there corresponds D', which contains a collection of relations M, each described by one of the types: a regular or weak entity relation, or a regular or weak relationship relation such that D' strongly preserves D under the collection of primary keys $(E \cup R)$. The converse if true.

Proof: It follows from Theorem 2.

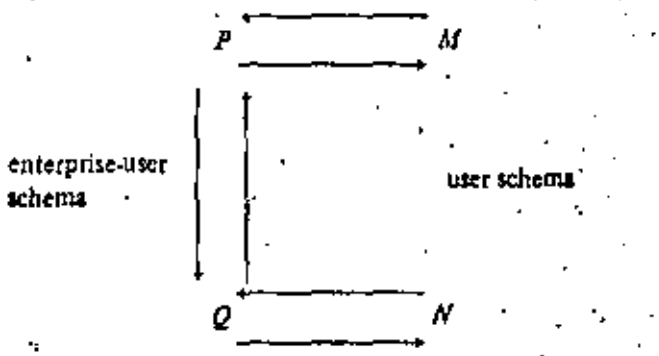
Corollary 3: For any NERR $M = (E, R, V, F)$, there corresponds a DRER, DWER, DRRR, or DWRR $M' = (K, R', V, F')$ such that M' strongly preserves M under $K(E \cup R)$. The converse is true.

Proof: In the proof of Theorem 2, it can be shown that the set of attributes is indeed a set of functions.

Corollary 4: For any NRER, NWER, NRRR, or NWRR $M = (K, R, V, F)$, there corresponds a DERR $M' = (E', R', V', F')$ such that M' strongly preserves M under $K(E \cup R)$ or $E \cup R$. The converse is true.

Proof: In the proof of Theorem 2, it can be shown that the set of attributes that we have constructed for the NERR are functions.

At this point, we can summarize our results, which we have accomplished, in terms of the following graph. Let M and N, respectively, be a nondeterministic and a deterministic relation, each described by one of the types: a regular or weak relation, or a regular or weak relationship relation. Let P and Q be a NERR and DERR, respectively. Let $M \rightarrow N$ denote that M can be transformed into N such that N strongly preserves M under $K(E \cup R)$, where K is used to identify uniquely entities in the entity set E or relationships in the relationship set R.



V. The Entity-Relationship Models Versus Other Data Models

In [2], [3], [13], the use of an entity-relationship diagram is fully explained, along with rules and examples for translation into hierarchical or network structures. In this section, we shall modify the major steps in logical database design using the entity-relationship approach in [3], [14]. The modified steps in logical database design can be used to derive entity-relationship relations that correspond equivalently to $3NF$ relations of the relational model. Hence, the use of entity-relationship diagram for translating into $3NF$ relations supported by the relational database system can be realized.

In the relational model, to illustrate transitive dependency, consider a relation scheme S concerning information about employees, departments where they work, and the nature of the work within the department [9].

$S(E\#, EN, JC, D\#, M\#, CT)$ where
$E\#$ = employee identification number,
EN = employee name,
JC = employee job code,
$D\#$ = department number of employee,
$M\#$ = identification number of department manager, and
CT = contract type (government or nongovernment).

In the description of the database application, the functional relationships among application attributes are given as follows.

Each employee is given only one job code and is assigned to only one department. Each department has its own manager and is involved in work on either government or nongovernment contracts, not both. The nontrivial functional dependencies in S are as shown in Fig. 14 (the nondependencies are implied). Clearly, S is not in $3NF$.

Using the process of further normalization, the relation scheme S can be converted to a relational schema containing $S1(\underline{E\#}, EN, JC, D\#)$ and $S2(\underline{D\#}, M\#, CT)$, which are in $3NF$. In addition, no essential information has been lost, since at any time, the original relations can be recovered by taking the natural join $S1$ and $S2$ on $D\#$.

Using the entity-relationship approach to the logical database design, without further considerations, we might define the enterprise schema for S, in terms of an entity-relationship diagram for EMP (employee) which is given in Fig. 15, and then translate this enterprise schema into a user schema, described by the regular entity relation EMP, which is given in Fig. 16. Clearly, all the undesirable anomalies, that can be created by insertion, updating, and deletion of an tuple in the relation scheme S, can also occur in the regular entity relation EMP. To eliminate these problems, the relation scheme S must be put into $3NF$. That is, the relation scheme S must satisfy the property: every nonkey attribute is fully functionally and nontransitively dependent on the primary key. In the regular entity relation, EMP of the ERD for EMP, the attributes which map from $E\#$ into their own value sets, are functional. The determinism of entity-relationship relations, described by the types: the regular or weak entity relations, or the regular or weak relationship relations, do not guarantee that the entity-relationship relations can overcome all anomalies with respect to storage operations. However, in Fig. 15, the entity-relationship diagram for EMP does not capture all the given semantics, namely, the given description of the database application in the real world. For instance, it does not include the fact that "each department has its own manager and is involved in work on either government or nongovernment contracts, not both." In conformity to the given description of the database application, a number of directed edges can be added deliberately to the value sets in the entity-relationship diagram for EMP. However, the obtained diagram given in Fig. 17 is no longer an entity-relationship diagram.

In order to conform to the semantics "each department has its own manager and is involved in work on either government or nongovernment contracts, not both," and to preserve the property of an entity-relationship diagram, a relationship set DEPT-MANAGER must be created. The relationship set is a one-to-one mapping defined on two entity sets DEPT and EMP. The relationship set has a functional attribute INVOLVED-PROJ-TYPE, which maps relationships of the relationship set DEPT-MANAGER into a value set CT (contract type). In Fig. 18, the entity-relationship diagram does conform to the functional dependencies, given in Fig. 14. It should be noted that the relationship set DEPT-EMP is a one-to-many mapping defined on two entity sets DEPT and EMP; this construct justifies the fact that DEPT is functionally dependent on EMP. The attribute INVOLVED-PROJ-TYPE maps functionally relationships of the relationship set DEPT-EMP into the value set CT.

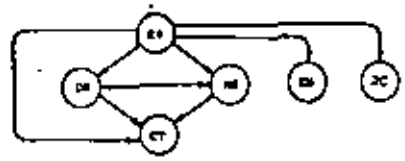


Fig. 14. Example of several transitive dependencies.

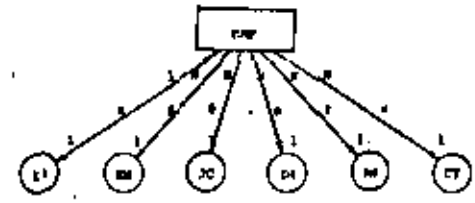


Fig. 15. An entity-relationship diagram for EMP, where EMP-ID is the primary key, α, β, δ, ρ, χ, and ε denote EMP-ID, EMP-NAME, JOBCODE, HIS-DEPT-NO, HIS-DEPT-MANAGER, and HIS-WORKING-PROJ-TYPE, respectively.

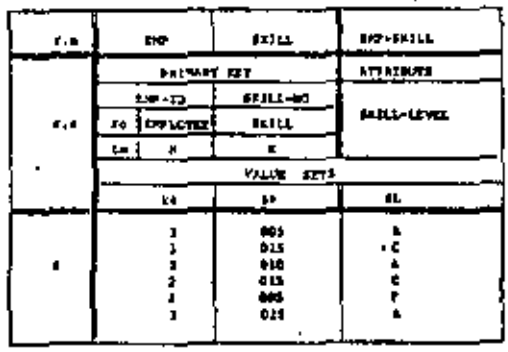


Fig. 16. A regular entity relation EMP.

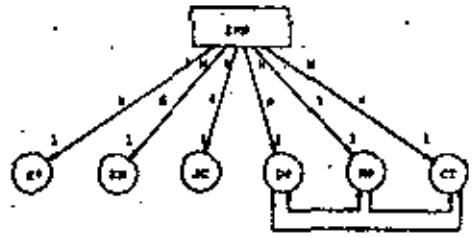


Fig. 17. A diagram conformable to the functional dependences given in Fig. 14.

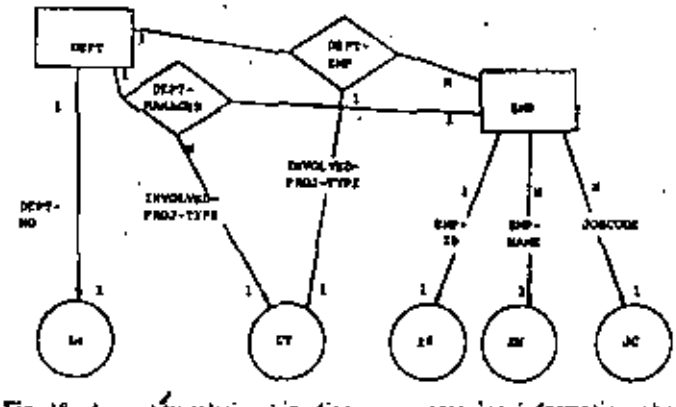


Fig. 18. An entity-relationship diagram concerning information about EMP and DEPT.

The construct also conforms to the fact that CT is functionally dependent on DEPT and is transitively dependent on EMP. However, the attribute INVOLVED-PROJ-TYPE, which maps functionally relationships in the relationship set DEPT-EMP

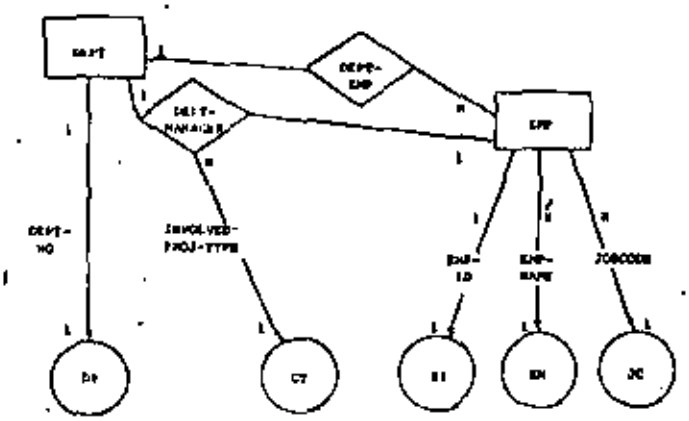


Fig. 19. An entity-relationship diagram concerning information about EMP and DEPT.

into the value set *CT*, is redundant because of the existence of the attribute INVOLVED-PROJ-TYPE which also maps functionally relationships in the relationship set DEPT-MANAGER. The existence of the attribute INVOLVED-PROJ-TYPE mapping relationships in the relationship set DEPT-MANAGER into the value set *CT* implies that *CT* is transitively dependent on EMP (because *CT* is functionally dependent on DEPT, and DEPT is functionally dependent on EMP) [4], [5]. The elimination of the redundant attribute INVOLVED-PROJ-TYPE yields the entity-relationship diagram as shown in Fig. 19, which conforms to the given semantics of the real world. Figs. 20–22 illustrate the translation from the entity-relationship diagram given in Fig. 19 into a user schema.

Several problems will arise from this construct.

1) *In Eliminating the Redundant Relationship Sets:* Without loss of generality, let the entity set EMPLOYEE be partitioned into two entity sets EMP′ and MEMP, containing nonmanagerial and managerial employees, respectively. Then, as shown in Fig. 23, create three relationship sets DEPT-EMP, DEPT-MANAGER, and EMP-MANAGER defined on the entity sets DEPT and EMP′, the entity sets DEPT and MEMP, and the entity sets EMP′ and MEMP, respectively. This construct justifies the facts that DEPT is functionally dependent on EMP′ (employee), that DEPT is functionally dependent on MEMP (manager), and vice versa, and that MEMP is functionally dependent on EMP′. That is, this entity-relationship diagram conforms to the given semantics of the real world: "Each employee is assigned to only one department. Each department has its own manager . . . ." However, the existence of the relationship set EMP-MANAGER defined on the entity sets EMP′ and MEMP is redundant, because the functional dependencies of DEPT on EMP′ and of MEMP on DEPT imply the functional dependency of MEMP on EMP′. Furthermore, the diagram shown in Fig. 23 is equivalent to the entity-relationship diagram shown in Fig. 24, which is obtained by combining the entity sets MEMP and EMP′ into EMP. The elimination of the relationship set EMP-MANAGER from both diagrams does not delete any semantic information from the original diagrams. Thus, in the entity-relationship diagram shown in Fig. 19, the functional dependency of MANAGER on EMP (nonmanagerial employee) is implicitly defined. Thus, the claim that the existence of the attribute INVOLVED-PROJ-TYPE mapping relationships in the relationship set DEPT-

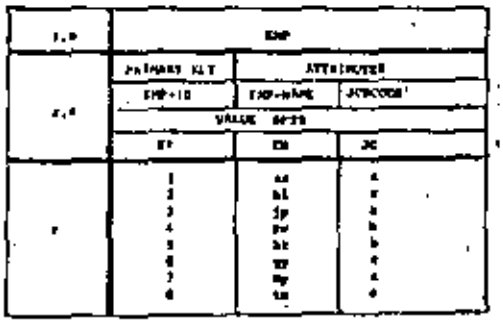


Fig. 20. A regular entity relation EMP.

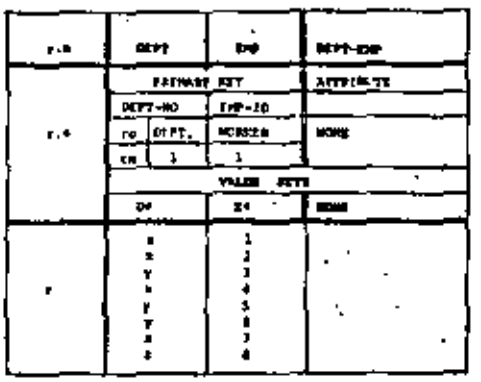


Fig. 21. A regular relationship relation DEPT-EMP. *1m* denotes type of mapping.

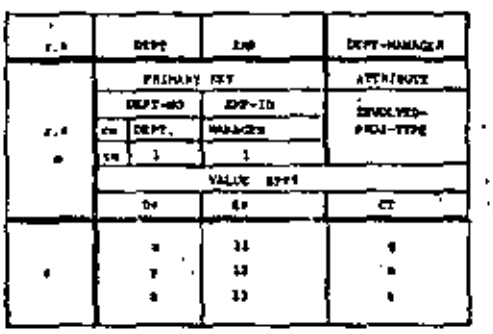


Fig. 22. A regular relationship relation DEPT-MANAGER.

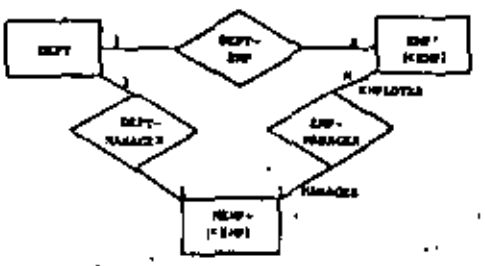


Fig. 23. An entity-relationship diagram.

MANAGER into the value set *CT* implies that *CT* is transitively dependent on EMP, is valid.

It should be noted that, in Figs. 23 and 24, instead of the relationship set EMP-MANAGER, the elimination of the relationship set DEPT-EMP also does not delete any semantic information from the original diagrams. However, the elimination of the relationship set DEPT-MANAGER from the diagrams will destroy, in turn, the given semantic information: "each department has its own manager . . . ." In the reality, the implication of functional dependence from others is not quite obvious, especially, if a very large set of functional de-

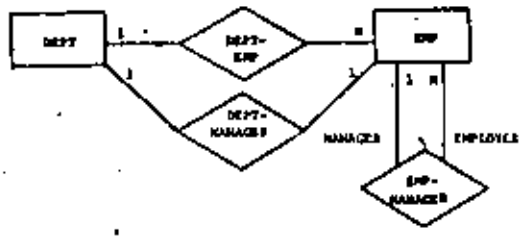


Fig. 24. An entity-relationship diagram.

pendencies is considered, and the criteria for determining which of the redundant relationship set is to be eliminated without modifying any given semantic information, is nontrivial.

In order to allow an additional illustration of full functional dependency, the description which concerns employees, their skills and skill levels [10] which is shown in Fig. 25, is added to the description of S which is given in Fig. 14. Fig. 25 represents the semantic information. "An employee may hold several skills (represented by the skill number $S\#$ and its skill name SN) and a particular skill may be held by several employees. A particular employee has different skill levels (represented by SL) for the skills he holds. A particular skill is held by several employees each of whom may have a different skill level for the skill in question." It is clear that SL is fully functionally dependent on the concatenated key $E\#.S\#$. As shown in Fig. 26, by creating the relationship set EMP-SKILL, the fully functional dependency of SL on $E\#.S\#$ can be expressed exactly and implicitly in the entity-relationship diagram. For the diagram shown in Fig. 26, the corresponding relations are given in Figs. 27 and 28.

2) *In Constructing the User Schema in Third Normal Form:* Consider the regular relationship relation DEPT-MANAGER. The corresponding relation scheme (in the relational model) is $S2(\underline{D\#}, \underline{M\#}, CT)$, in which only $D\#$ or $M\#$, but not both, is to be selected as the primary key because of the fact that the mapping between DEPT-NO/DEPARTMENT and EMP-ID/MANAGER is a one-to-one correspondence [10]. This is one of the major reasons that the attribute INVOLVED-PROJ-TYPE which maps relationships into the value set CT is kept away from the elimination. However, it is true that if both $D\#$ and $M\#$ are primary keys, the $S2$ is no longer a relation of third normal form, because CT is not fully functionally dependent on ($D\#$, $M\#$). That means, the regular relationship relation DEPT-MANAGER is not similar to a $3NF$ relation, even if the mapping between DEPT-NO/DEPARTMENT and EMP-ID/MANAGER is one-to-many, many-to-one, or many-to-many.

Consider again the entity-relationship diagram given in Fig. 19, in which the attribute INVOLVED-PROJ-TYPE maps relationships in the relationship set DEPT-MANAGER into the value set CT. That means, each of the entity pairs of the entity sets DEPT and EMP/MANAGER is involved in work on either government or nongovernment contracts, not both. Since the relationships DEPT-MANAGER over the entity sets DEPT and EMP/MANAGER are one-to-one correspondence, in order to conform to the given semantic information, it suffices to define the attribute INVOLVED-PROJ-TYPE mapping the entity set DEPT into the value set CT, as shown in Fig. 29. One of the major differences of this connection of the value set CT to the entity set EMP from that in Figs. 18 and 19 is that the latter

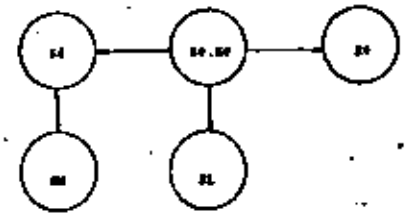


Fig. 25. A description concerning employees, their skills, and skill levels.

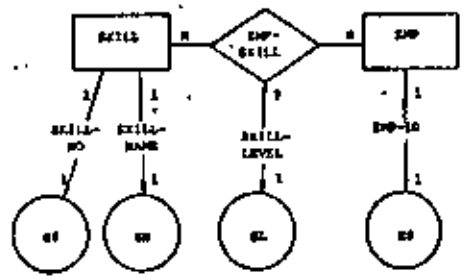


Fig. 26. An entity-relationship diagram.

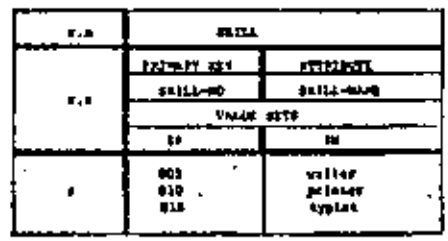


Fig. 27. A regular entity relation SKILL.

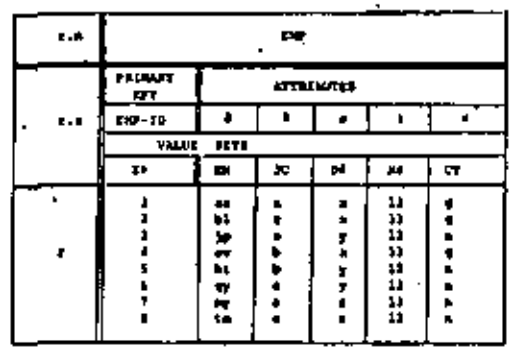


Fig. 28. A regular relationship relation, EMP-SKILL.

connections do not satisfy the fully functional dependency, and, therefore, the problem that the regular relationship relation DEPT-MANAGER is not in $3NF$ could be avoided. Fig. 30 is a modified regular relationship relation DEPT-MANAGER, which is in $3NF$. As a matter of fact, it is a nontrivial problem of deciding what are the value sets and attributes for the entity or relationship sets from the given description of an application. For example, the value set CT and the attribute INVOLVED-PROJ-TYPE are assigned to the relationship set DEPT-MANAGER is indeed an erroneous representation.

Summarily, the user schema in the entity-relationship model, as shown in Figs. 20 and 21, Figs. 27 and 28, and Figs. 30 and 31 obviously are similar to $3NF$ relations in the relational model, but the former relations have clearer representation, and are obtained without using the normalization operations [4], [5], [9]. In the normalization of the relational model,

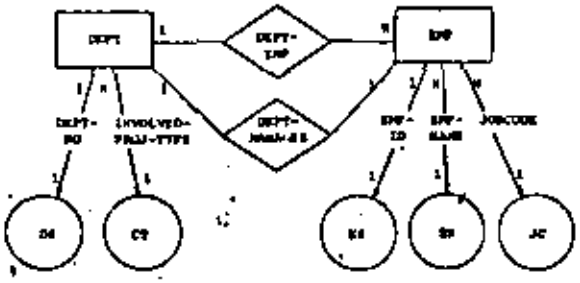


Fig. 29. An entity-relationship diagram concerning information about EMP and DEPT.

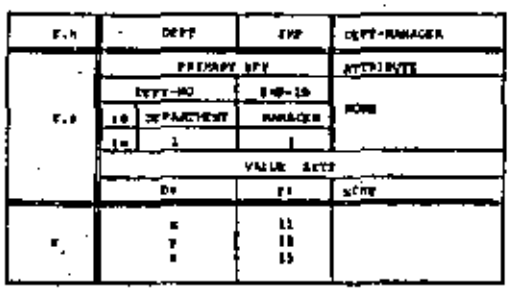


Fig. 30. A modified regular relationship relation, DEPT-MANAGER.

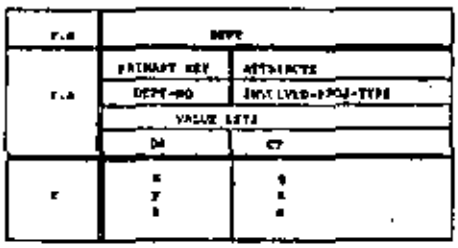


Fig. 31. A regular entity relation DEPT.

$S1(\underline{E\#}, EN, JC, D\#)$, $S2(\underline{D\#}, M\#, CT)$, $S3(\underline{E\#}, S\#, SL)$, and $S4(\underline{S\#}, SN)$, are the fewest possible number of 3NF relations [4], [5], [9]. Perhaps, the severe setback for the entity-relationship approach is the tradeoff between the clearer semantic representation and the fewest possible number of relations.

Consider again the diagram given in Fig. 16. For the entity set EMP, implicitly, the value set $D\#$ is functionally dependent on $E\#$ (the primary key), but not vice versa. It is easy to show that the nonkey value set CT is transitively dependent on the primary key value set $E\#$. However, such a transitive dependency can be eliminated by introducing a relationship set DEPT-EMP, as shown in Figs. 18, 19, and 29. For each of the entity sets EMP and DEPT and the relationship sets DEPT-EMP and DEPT-MANAGER, no transitive dependency among value sets is present. Therefore, the use of the entity-relationship approach is eliminating the transitive dependency among value sets is more effective than the normalization process for the relational model. Furthermore, by the definition of attribute, the entity-relationship diagram always provides the fact that each nonkey value set is fully functionally dependent on the primary key's value set(s).

Now, we shall outline the major steps in the logical database design using the entity-relationship approach. Given a description of an application for an enterprise, we intend to find a "good" schema that describes the structure of the corresponding database. Unfortunately, the entity-relationship approach, which we shall outline as follows, is a heuristic approach. In addition, we are only considering the deterministic case, for the sake of simplicity.

The entity-relationship approach to the logical database design consists of the following major steps.

1) *Given a description of an application.* The description of a database application for an enterprise can be formulated as a set of functional relationships among application attributes. Whether the set of functional relationships among application attributes is complete, nonredundant, and minimal [4], [5] is of no concern.

2) *Identify entity sets.* From the given description of an application, identify all possible entity sets which are of interest to the enterprise.

3) *Identify relationship sets.* Relationships may exist among the entities, and can be classified by the relationship sets. In general, there are m-way relationships defined on entity sets, where m is greater than or equal to 1. Conformable to the given description of an application, identify all possible relationship sets and specify their types of mapping (e.g., one-to-one, one-to-many, or many-to-many). Eliminate the redundant relationship sets.

4) *Draw an entity-relationship diagram with entity and relationship sets.*

5) *Identify value sets and attributes.* From the given description of an application, we may decide what are the value sets and attributes for the entity or relationship sets. The functional attributes, mapping from the entity or relationship sets into the value sets or the Cartesian product of the value sets must conform to some functional dependencies among application attributes. At the end of this step, an entity-relationship diagram with the entity and relationship sets, the value sets, and the attributes are obtained.

6) *Identify primary keys for entity sets.*

7) *Check the entity-relationship diagram in conformity to the description of the application.* More specifically, does the entity-relationship diagram include all possible functional relationships among the value sets? Is the transitive dependency among application attributes "properly" represented in the entity-relationship diagram? Is every nonkey attribute value set(s) nontransitively or fully dependent on the primary key value set(s)? If the answer of any of these questions is inaffirmative, then go back to step 2).

8) *Translate the entity-relationship diagram into entity-relationship relations.* These relations are described by any one of the deterministic types: the regular or weak entity relations, or the regular or weak relationship relations.

For any entity-relationship diagram, it also can be translated into data-structure diagrams supported by the CODASYL (network) type database system or hierarchical structures supported by a hierarchical database system such as IBM's IMS [2], [3].

Although it can be translated into the relational schema supported by the relational database system, it would be easier to obtain the relational schema by transforming directly from the entity-relationship relations, described by the regular or weak entity relations, or the regular or weak relationship relations.

It should be noted that if one wants to change from one

nondeterministic relation to a deterministic relation, one would probably have to change the enterprise-user schema and then reconstruct the user schema, since the enterprise-user schema using the entity-relationship relations and the enterprise schema using the entity-relationship diagram are independent of the primary keys, the unique identification of entities, and relationships. The changes of the enterprise-user schema and enterprise schema are one-to-one corresponding, and, thus, the conformity between the enterprise schema and the description of the real world can be checked.

9) *Design record formats.*

From our discussion, clearly, we can state the following theorem.

Theorem 3: For any entity-relationship relation M, described by one of the types: the regular or weak entity relation, or the regular or weak relationship relation such that none of the nonkey attribute value set (or Cartesian product of value sets), which is fully dependent on the primary key value set(s), is transitively dependent on the primary key value set(s), then M can be transformed into a $3NF$ relation (in the relational model) that strongly preserves M under the primary keys.

VI. Physical Representation of Logical Structures

In this section, we will discuss physical representation of logical structures. In the previous sections, we are concerned with the way the database administrator or those system analysts who see the entire database and the application programmer views of data. Neither the entity-relationship diagram nor the corresponding relations reflects the way the data are stored physically. In the sequel, we will be concerned with how it is laid out on the storage units. That is, for a given overall view of the data in terms of the schemas, how do we represent this overall logical organization physically.

Given an entity-relationship diagram, it can be translated into a data-structure diagram [5], [6] by the following rules.

1) For each of the entity (relationship) sets in the entity-relationship diagram, there corresponds an entity (relationship) record type, called an owner (member) record type, in the data-structure diagram.

2) For $1:N$ binary relationships of a relationship set C defined on two entity sets A and B, in the entity-relationship diagram, two directed links, one with double arrows and the other with a single arrow, are drawn from the entity record types A and B, respectively, to the relationship record type C.

For $M:N$ binary relationships, two directed links with double arrows are drawn.

Likewise, for k-ary ($k \geq 3$) relationships, the same rules apply.

3) If the existence and identification of an entity set B are dependent on another entity set A in the entity-relationship diagram, then a path dependency from the entity record type A to the entity record type B is created in the data-structure diagram. The path dependency can be represented by a directed line with single or double arrows pointed to B depending upon whether the mapping from the entity set A into the entity set B is of one or many associations, respectively.

If only the existence dependency of B on A occurs, then the path dependency can be represented by a bidirected line, which can be obtained by adding a single arrow (pointed to A) on the other end of the directed line.

Fig. 32 is a data-structure diagram which corresponds partially to the entity-relationship diagram given in Fig. 1. The structure demonstrates the $M:N$ relationship set E and EMP-CHILD on the entity sets EMPLOYEE and CHILDREN, and the $M:N$ relationship set E and I (the existence and identification dependencies) on the entity set EMPLOYEE and PARENTS.

Clearly, the entity relationship model can be used as a tool in the structured design of databases using the network models [6]. The user first draws an entity-relationship diagram and then may simply translate it into a data-structure diagram using the rules specified above.

In the remaining section, we will use the entity-relationship diagram shown in Fig. 1 to demonstrate the design of the record formats, and the use of pointers for representing associations between records or segments and associations between data.

In the course of constructing the physical representation of the entity-relationship diagram, the corresponding data-structure diagram and the corresponding relations can be used as major components of an integrated design tool to assist in the physical database design process. However, it suffices to use the entity-relationship diagram and/or the corresponding relations which are described by one of the types given in the previous section as a design tool in designing the physical database organization.

The data-structure diagram such as that in Fig. 32, can be represented physically in a variety of ways. Figs. 33-35 illustrate one of these ways, which involves parent, child, twin, and cousin pointers. Fig. 34 is a physical representation of the data-structure diagram of the record types PROJ-EMP, EMPLOYEE, and PROJECT. In the relationship record type PROJ-TYPE, two parent pointers are used from each relationship record to that record's parents, which are of entity record types EMPLOYEE and PROJECT. From these two parent pointers, the primary keys of each of the relationships can be accessed directly from the entity record record types EMPLOYEE and PROJECT.

For the existence and identification dependencies of the entity set PARENTS on the entity set EMPLOYEE, as shown in Fig. 34, a relationship record type is used to represent physically the existence dependency and a cousin pointer from each instance of the entity record type PARENTS to the instance of the subordinate entity record type is used to represent the identification dependency. That is, each instance of the entity record type PARENTS is identified by the combination of the primary keys, one from the EMPLOYEE and the other from itself, PARENTS, as shown in Fig. 12. If only the existence dependency occurs, then the cousin pointer will not be used.

In Fig. 35, all the instances of the relationship record type MARRIAGE contain two physical parent pointers. However, both parent pointers of each of the instances are pointing to the different instances of the same entity record type PARENTS, since the relationship set MARRIAGE is defined on a singleton set, the entity set PARENTS. Again, each of the relationships is identified by the combination of the primary keys, one from

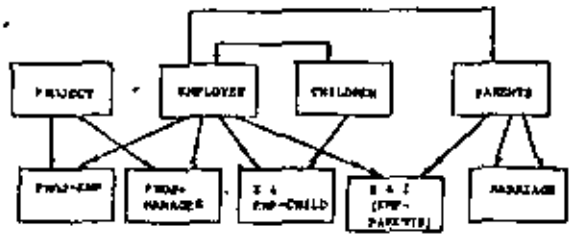


Fig. 32. A logical data-structure which corresponds partially to the entity-relationship diagram given in Fig. 1.

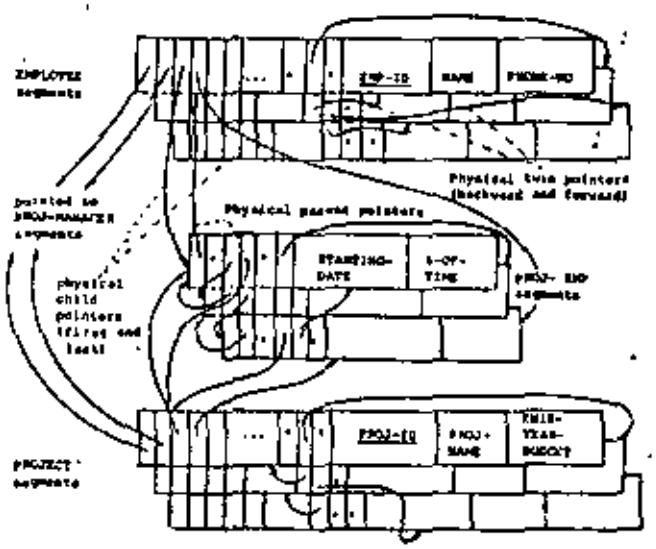


Fig. 33. A physical representation of the data-structure diagram of the record types PROJ-EMP, EMPLOYEE, and PROJECT.

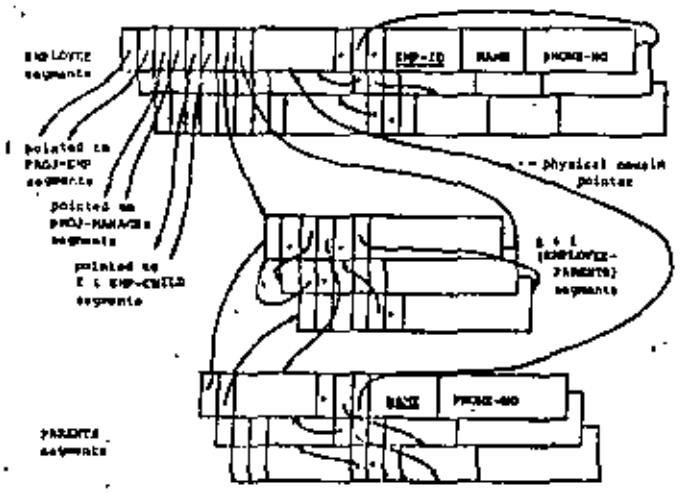


Fig. 34. A physical representation of the data-structure diagram of the record types *E* and *I* (EMPLOYEE-PARENTS), EMPLOYEE, and PARENTS.

the entity set EMPLOYEE and the other from itself, PARENTS, as shown in Fig. 13, the weak relationship relation PARENTS-MARRIAGE-RECORD.

VII. Conclusion

In this paper, we have defined formally the entity-relationship model in terms of the entity-relationship diagram as the enterprise schema, the entity relationship relations as the enterprise-user schema, and four types of relations as the user schema. We have described the entity-relationship approach to logical database design. The fact that the nondeterministic

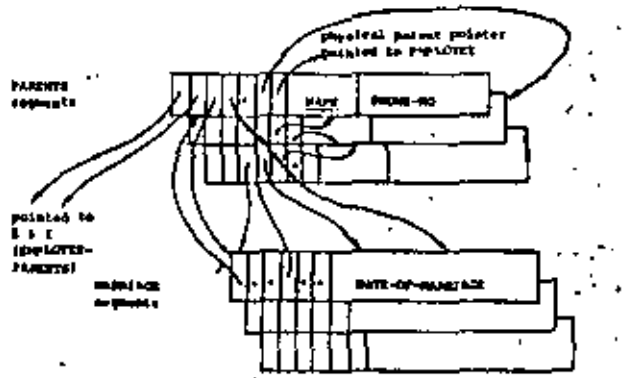


Fig. 35. A physical representation of the data-structure diagram of the record types MARRIAGE and PARENTS.

and deterministic have the same strength to characterize information about the real world enhances the entity-relationship approach because the database designer is not constrained by the functional attributes to identify the properties of entities and their relationships.

The entity-relationship approach which provides an easy to understand yet comprehensive methodology for the logical database design independent of storage and efficiency considerations is to add to the logical database design process, an intermediate stage between information about entities in the real world and the user schema, the final product of the process. The product of this intermediate stage is the enterprise schema, the description of an enterprise view of data in the real world in terms of entity-relationship diagrams. In this paper, we have explored that the entity-relationship diagram can be used as an effective tool in describing exactly and concisely the properties of entities and their relationships, in eliminating the redundant relationships among the entities and in constructing the user schema in third normal form without using the process of normalization. In addition, we have presented an improved table form of the relations as the user schema to provide a clear, and concise, and better organized user's view of databases. It is easy to understand and to modify.

Finally, we have demonstrated the process of physical representation of logical structures. The construction of the soundness and completeness of an entity-relationship approach to the logical database design requires further research. It seems desirable that a theoretical study of properties of the model be conducted in the framework of formalism.

Acknowledgment

The author is grateful to P. P. S. Chen and R. T. Yeh for their comments on an earlier version of this paper.

References

[1] E. F. Codd, "A relational model of data for large shared data banks," *Commun. ACM*, vol. 13, pp. 377–387, June 1970.

[2] P. P. S. Chen, "The entity-relationship model: Towards a unified view of data," *ACM Trans. Database Syst.*, vol. 1, no. 1, pp. 9–36, 1976.

[3] P. P. S. Chen, "The entity-relationship approach to logical database design," Q.E.D. Information Science, Inc., Wellesley, MA, 1977.

[4] P. A. Bernstein, "Synthesizing third normal form relations from functional dependencies," *ACM Trans. Database Syst.*, vol. 1, no. 4, pp. 277–298, 1976.

[5] C. Beeri and P. A. Bernstein, "Computational problems related to the design of normal form relational schemas," *ACM Trans. Database Syst.*, vol. 4, no. 1, pp. 30–59, 1979.
[6] C. J. Date, *An Introduction to Database Systems*. Reading, MA: Addison-Wesley, 1975.
[7] R. Fagin, "The decomposition versus synthetic approach to relational database design," in *Proc. 3rd Int. Conf. VLDB*, Tokyo, Japan, October 1977, pp. 441–446.
[8] C. Delobel and R. G. Casey, "Decomposition of a database and theory of boolean switching functions," *IBM J. Res. Develop.*, vol. 17, pp. 374–386, Sept. 1972.
[9] E. F. Codd, "Further normalization of the database relational model," in *Data Base Systems, Courant Computer Science Symposia Series*, vol. 6. Englewood Cliffs, NJ: Prentice-Hall, 1972, pp. 33–64.
[10] M. Vetter, "Database design by applied data synthesis," in *Proc. 3rd Int. Conf. VLDB*, Tokyo, Japan, Oct. 1977, pp. 428–440.
[11] Y. E. Lien, "On the semantics of the entity-relationship model," in *Entity-Relationship Approach to Systems Analysis and Design*, P. P. S. Chen, Ed. Amsterdam, The Netherlands: North-Holland, 1980, pp. 155–168.
[12] M. A. Malkanoff and C. Zaniolo, "Decomposition of relations and synthesis of entity-relationship diagrams," in *Entity-Relationship Approach to Systems Analysis and Design*, P. P. S. Chen, Ed. Amsterdam, The Netherlands: North-Holland, 1980, pp. 277–294.
[13] H. Sakai, "A unified approach to the logical design of a hierarchical data model," *Entity-Relationship Approach to Systems Analysis and Design*, P. P. S. Chen, Ed. Amsterdam, The Netherlands: North-Holland, 1980, pp. 61–74.
[14] P. A. Ng and J. F. Paul, "A formal definition of entity-relationship model," *Entity-Relationship Approach to Systems Analysis and Design*, P. P. S. Chen, Ed. Amsterdam, The Netherlands: North-Holland, 1980, pp. 211–230.
[15] *Entity-Relationship Approach to Systems Analysis and Design*, P. P. S. Chen, Ed. Amsterdam: The Netherlands: North-Holland, 1980.

Peter A. Ng received the B.Sc. degree in mathematics from St. Edward University, Austin, TX, in 1969, and the Ph.D. degree in computer sciences from the University of Texas, Austin, in 1974.

He served two years as Assistant Professor of Computer Sciences at Hunter College, City University of New York, New York, NY, from 1975 to 1976. In August 1976 he joined the Department of Computer Science, University of Missouri, Columbia, where he is presently an Associate Professor. His current major research interests are information systems, database management system, data communications, and several aspects of software engineering. He has published over 30 technical papers on these fields in international journals and conference proceedings. He is an industrial Consultant on these areas.

Dr. Ng is a member of the Association for Computing Machinery and the IEEE Computer Society.

Mapping Considerations in the Design of Schemas for the Relational Model

SABAH AL-FEDAGHI AND PETER SCHEUERMANN

Abstract—The typical design process for the relational database model develops the conceptual schema and each of the external schemas separately and independently from each other. This paper proposes a new design methodology that constructs the conceptual schema in such a way that overlappings among external schemas are reflected. If the overlappings of external schemas do not produce transitivity at the conceptual level, then with our design method, the relations in the external schemas can be realized as a join over independent components. Thus, a one-to-one function can be defined for the mapping between tuples in the external schemas to tuples in the conceptual schema. If transitivity is produced, then we show that no such function is possible and a new technique is introduced to handle this special case.

Index Terms—Conceptual schema, external schema, functional dependencies, independent components, interferences, logical database design, mapping functions, relational database mode.

Manuscript received February 1, 1980; revised June 11, 1980. This work was supported in part by the National Science Foundation under Grant MCS77-03904.

The authors are with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60201.

I. Introduction

IN THE RELATIONAL database model, a given external view V_k and the conceptual view V are described by sets of relations. The set of relations in the conceptual view constitutes the *conceptual schema* and the set of relations in a given external view constitutes an *external schema*. Given a relational model let $E = \{E_1, E_2, \cdots, E_n\}$ be the set of the external schemas and C be the conceptual schema of the model. The mapping problem covers several transformations between a given $E_k \in E$ and C. Let $E_k = \{e_1^k, e_2^k, \cdots, e_v^k\}$ and $C = \{c_1, c_2, \cdots, c_u\}$ where e_i^k, $1 \leq i \leq v$, is a relation in E_k and c_j, $1 \leq j \leq u$, is a relation in C. Several mappings are of interest.

1) *Relations Mapping:* $\alpha_1(e_i^k) = C' \subseteq C$. That is, α_1 constructs the relation $e_i^k \in E_k$ from the relations C' in C.

2) *Tuples Mapping:* $\alpha_2(t(e_i^k))$, where $t(e_i^k)$ is a tuple in e_i^k. The function α_2 maps a tuple of e_i^k to a tuple or a set of tuples in $C' \subseteq C$.

0098-5589/81/0100-0099$00.75 © 1981 IEEE

DISEÑO DE BASES DE DATOS

B S P CONCEPTS

EXPOSITOR:
ING. DANIEL RIOS ZERTUCHE

MAYO, 1984

Chapter 1. BSP Concepts

Business Systems Planning is most often thought of as a structured approach or methodology. This methodology, however, is based upon some fundamental concepts, a good understanding of which can give the BSP study team members:

1. A better appreciation of the "why's" of the methodology
2. Improved confidence in applying variations to meet specific situations
3. A better background with which to communicate the objectives and eventual recommendations to senior management

The premise for conducting a BSP study is that there exists within the organization a need for significantly improved computer-based information systems (I/S) and a need for an overall strategy to attain them. BSP is concerned with how these information systems should be structured, integrated, and implemented over the long term. The basic concepts of BSP can be related to the long-term objectives for I/S in an organization.

An I/S Must Support the Goals and Objectives of the Business

This most basic concept underlies the "top down" philosophy of the methodology as well as several of the specific steps, such as executive interviews and system priorities.

Since information systems can be an integral part of a business and be critical to its overall effectiveness, and because they will continue to represent major investments of time and money, it is essential that they support the organization's true business needs and directly influence its objectives. BSP, then, can be thought of as a vehicle or process to translate business strategy into I/S strategy (see Figure 1).

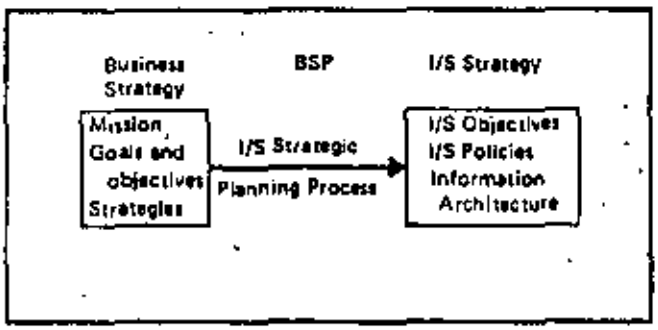


Figure 1. Translation of business strategy to I/S strategy

Obviously, it is important that an organization be willing and able to express its long-term goals and objectives. For some organizations, this can be done principally through the business plan. For others, where a business plan is not available or current, it can be done as a part of the BSP methodology. In either event, a recognition of this basic need by senior management is critical, for only with that recognition will their commitment and involvement be great enough to guarantee a meaningful BSP study.

An I/S Strategy Should Address the Needs of All Levels of Management Within the Business

This requirement has several implications relative to I/S structure. First, it is important to recognize the varying characteristics of information as needed by different activities and management levels. Typically, lower levels need considerable detail, volume, and frequency, higher levels need summaries, "exception" reporting, and inquiries, and still higher levels need cross-functional summaries, special requests, "what if" analyses, "external" requirements, etc. It would be impractical to construct a single system to accommodate all activities or management levels, and it would be erroneous to associate any one type of information requirement solely with one management level. Clearly, there is need to establish some reasonable framework upon which the I/S can be defined.

First, the emphasis in I/S should be in support of management decision making. This is in contrast to more traditional bookkeeping or recordkeeping functions. Business decisions are made for various purposes, but most can be associated with either *planning* or *control*. Planning, of course, is the establishment of various missions, objectives, and policies, and it occurs at all levels. Good information is vital to the establishment of good plans. Control decisions, on the other hand, are made in order to guide an activity toward some implicit or defined (by the plan) objective. The I/S can provide the measurements of the current or actual condition to the decision maker. We thus complete a planning, measurement and control cycle with I/S potentially an integral part. Since planning and control are the keys to decision making, a framework for I/S based upon these activities can be utilized. It has been proposed, * and well accepted today, that three distinct but concurrent planning and control levels exist in any organization:

> *Strategic planning*, the process of deciding on objectives of the organization, on the resources used to attain these objectives, and on the policies that are to govern the acquisition, use, and disposition of resources

*Anthony, R.N. *Planning and Control Systems: A Framework for Analysis*, Harvard Business School, Division of Research, 1965.

Management control, the process by which managers assure that resources are obtained and used efficiently in the accomplishment of the organization's objectives

Operational control, the process of assuring that specific tasks are carried out effectively and efficiently

Further characteristics of these areas are outlined in Figure 2. An advantage of this framework is that it does not restrict planning and control activity to any particular industry, function, or management level. A conclusion at this point is that an I/S could conveniently address itself to any one of the above three planning and control levels.

Resource management is also key to this philosophy and represents a major vehicle for I/S definition. The specific resources to be managed vary in nature and relative importance from one organization to the next. Examples of traditional resources to be managed are people, facilities, materials and money. Their requirements are based upon the needs to support the prime mission area of the organization, e.g., its products or services. Because an organization's product or service area has all the attributes of a resource, i.e., a life cycle of activities and decision points, and yet drives the other resources, it is referred to as the "key resource." Each resource, including the key resource, is managed through planning and control decisions of the three levels previously discussed. Resource management has the desired characteristic of cutting across organizational boundaries – vertically across management levels and horizontally across functional lines. Thus a framework based on resources as well as planning and control levels can be established, and an I/S architecture can be applied within this framework.

An I/S Should Provide Consistency of Information Throughout the Organization

The keyword in this objective is *consistency*. The implication is that information derived from more traditional data processing applications is not necessarily consistent, particularly when applied to new business problems (decision areas) of broader scope. The problems in data consistency normally arise as a result of a historical evolution of computer usage that traditionally occurs. Isolated and independent application areas are selected and mechanized, typically to reduce operational costs. The data files are defined as necessary to support the specific needs of each application without regard to one another or to future applications. The data itself is converted from manual files located and maintained by the using organization.

As computer applications are added, new data files are usually required since the data requirements for different applications are rarely the same. These are usually created from spinoffs of existing mechanized files plus any additional data that may be required from the using area. Summary-type reports for higher management levels are the result of sorting and merging various existing data files together to create new ones. Rarely is any existing data file of the form or content required to provide newly requested information of any magnitude. Thus new data files are born. Data redundancies and file update requirements multiply. The continual cry from management that "I know the data that I need is somewhere in data processing, but I can't get at it" may be basically true from their point of view. The data may be there, but not necessarily defined as needed, of adequate summary level or sequence, or of proper time period or currency.

Decision Characteristic	Planning and Control Level		
	Strategic Planning	Management Control	Operational Control
Management involvement	General management Functional management	General management Functional management Operational management	Functional management Operational management
Time horizon	Long range (1–10+ years)	Year-to-year Monthly	Day-to-day Weekly
Degree of structure	Unstructured and irregular; each problem different	More structured, cyclic, largely repeating	Highly structured, repetitious
Data requirements	Summaries, estimates, difficult to pre-define, much external to business	Summaries, definable, need for unanticipated forms, largely internal	Detail, operational, definable, internally generated
Resource management	Establishment of policies pertaining to the resource	Allocation of the resource	Efficient use of the resource

Figure 2. Characteristics of planning and control levels

Data, then, exists in most organizations in varying *form*, *definition*, and *time*. The *form* may be uncaptured raw data, mechanized data files, detailed DP reports, summarized DP reports, business documents, or knowledge in someone's head. The *definition* of any given data can have as many variations, and thus inconsistencies, as there are users of that data. For example, "salary" to the payroll department may mean an employee's actual monthly pay, to the project manager an annual figure plus burden to be charged to a customer, to the department manager a budget line item representing total expense for all reporting employees. In addition, a *time* inconsistency is very likely to exist between what may otherwise be comparable data. Data may be captured by such varying methods as the mail, telephone, data terminals, or satellites. It may be "batched" over varying lengths of time by the user or by DP before processing, or it may be entered "online" as each transaction occurs, directly from its source. Data may be processed daily, weekly, or monthly on a predetermined schedule, or it may be incapable of being processed until a series of prior computer runs are completed (e.g., the month-end closing). The output itself may be mailed or it could be immediately available as the result of an online inquiry. Finally, the report may be up to the second, as when reading from a terminal, or it may have lain in a desk for three weeks after last month's processing. The combinations of time deviations between data capture, data processing activity, and actual data usage are many.

With all these potential data inconsistencies, it is no wonder that reports frequently don't "match" between using departments and managers. This becomes a problem most often during interdepartmental decision-making or at higher reporting levels where consolidation of multifunction activities is important. Attempts to provide better data consistency usually result in "resystematizing" or "consolidating" existing applications into larger ones with broader problem scope and data definitions. This may yield a satisfactory system within the defined scope, but again, as still broader problems are addressed, there will undoubtedly be data inconsistencies between the larger systems. Resystematizing at this scope may be extremely expensive and difficult to justify, let alone accomplish. Comments prevail such as "our systems cannot talk to one another."

What has been described is the classical "bottom up" evolution of data processing systems. In order to begin to address the data consistency problem, a different philosophy must be adopted relative to data management. This is commonly referred to as *managing data as a resource*. This concept suggests that data is of considerable overall value to an organization and should be managed accordingly. It should be potentially available to and shared by the total business unit on a consistent basis. It should not be controlled by a limited organizational segment but by a central coordinator, much like other corporate resources, such as cash and personnel. The management function would include formulating policies and procedures for consistent definition, sourcing, technical implementation, use, and security of the data.

An I/S Should Be Able to Live Through Organizational and Management Change

Many data processing systems and applications are set up to provide the information needs of a specific department or other organizational entity. Others are built solely on the specific output report requirements of a particular manager. Both types can become immediately obsolete upon a reoganization or management change. A new manager may have his own ideas as to what information is needed to run the department. While this kind of change is inevitable, it can be expensive from a data processing standpoint. The data processing system, however, should in no way inhibit management flexibility in a dynamic enterprise. Thus, the I/S must anticipate and be capable of living through the long-term organizational and management changes of a business with minimum impact if the expected return on investments is to be realized.

This objective cannot be realized without the proper support vehicle for I/S, and this vehicle must be independent of the various components of the organizational structure. The BSP vehicle is the *business process*, that is, a basic activity and decision area irrespective of any reporting hierarchy or specific management responsibility. A logical set of these processes can be defined for any type of business and will undergo minimum change as long as the product or service area of the business remains basically the same.

One example of a business process is *purchasing*. A particular business might define this as "the process by which raw materials are acquired from vendors." There may or may not be a separate organizational unit to accomplish this process, or indeed there may be several. Inherent within this process are the various activities and decisions necessary to accomplish the process.

Defining the organization's business processes is one of the most important parts of the BSP methodology, and the method for doing so is tied directly to the previously discussed I/S framework, that is, one based on resources and planning and control levels. With this in mind it is convenient to define an organization's business processes in association with each of its defined resources. Emphasis in BSP is normally placed

upon those processes necessary to manage the *key resource*. Each resource of a business can be thought of as having a "life cycle" made up of several stages. A product life cycle, for example, has four stages: requirements, acquisition, stewardship, and retirement. The time spread of the life cycle can vary greatly with the particular product area but is of no consequence in this approach. Business processes can be identified to describe the major activities performed and decisions made by the business in the course of managing the resource throughout its life cycle. These can normally be organized into a process hierarchy, and this is done without regard to organizational involvement or responsibility.

The above approach results in process definitions that encompass the three planning and control levels previously discussed -- namely, strategic planning, management control, and operational control. Using a product resource as the example again, the decision to pursue a particular product area would be "strategic planning," the planning and control decisions relative to product volumes or advertising expenditures would be "management control," and the decisions in areas of engineering control, manufacturing efficiency, etc., would be "operational control." By using this approach for all the resources it is possible to define all the business processes that take place within any organizational segment. This may be tempered by practicality in the actual BSP as there may be little I/S support interest for some of the resources.

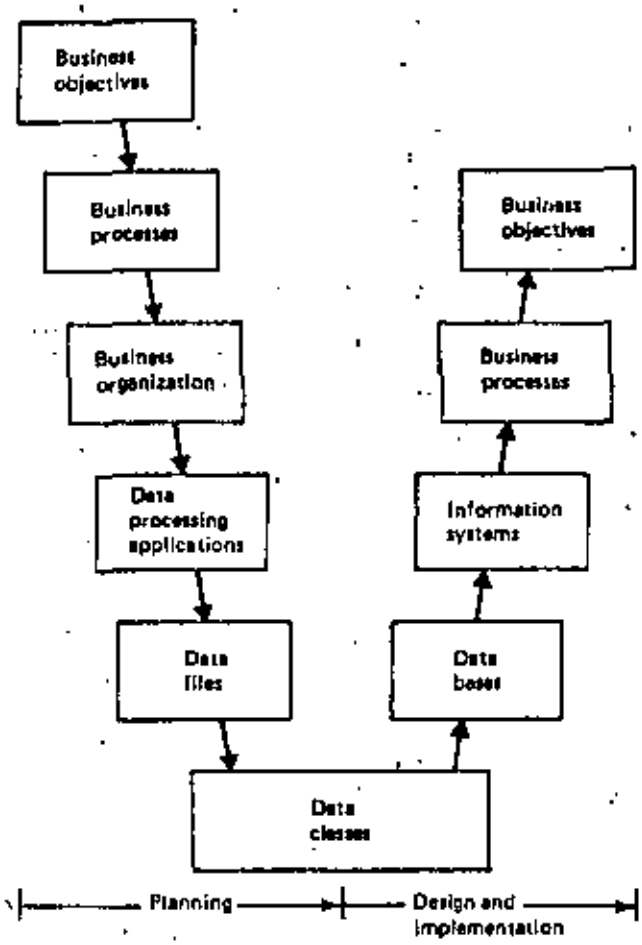


Figure 3. Top-down analysis with bottom-up implementation

The I/S Strategy Should Be Implemented By Subsystem Within a Total Information Architecture

There are several implications and concepts associated with this statement. The first is that a total I/S to support the entire business unit's needs is too big to build in any single project. However, because of the many problems associated with a "bottom up" evolution of systems (data inconsistencies, non-integrated system designs, expensive resystematizing, priority difficulties, etc.), it is very important that long-range goals and objectives for I/S be established. The basic concept, then, is *top-down I/S planning with bottom-up implementation* (Figure 3).

With this concept the long-range I/S goals and objectives are identified through a top-down planning process (the BSP approach). The identified systems are then implemented in a modular building-block fashion over time while remaining consistent with the organization's business priorities, available funds, and other shorter-term considerations. This philosophy can be likened to the detail design and construction of a large office building, which would be unthinkable without an architect's approved drawing of the finished product.

The BSP methodology, although consisting of considerably more steps and detail than shown in Figure 4, is consistent with this philosophy. Step 1 of Figure 4, defining the business objectives, is intended to ensure agreement among all executive levels as to where the business is going, so that the I/S strategy can be in direct support. Step 2, defining the business processes, establishes the prime long-term basis for I/S support in the business. Step 3, defining the data classes, can be done on the basis of the processes to be supported. A *data class*, as the name implies, is a major category of data needed to support one or more business processes. For example, customer information is a data class needed in several process areas, such as order entry, billing, and distribution. This step results in a definition of all the data to be managed as a resource across the business unit. Step 4, defining the information architecture, becomes a statement of the long-term I/S objective. This is normally in the form of a group of interrelated I/S areas and the associated data to be managed. From the information architecture the individual modules can be identified, prioritized, and built as scheduled by the I/S plan.

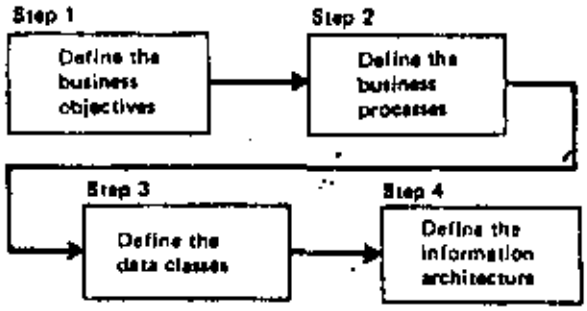


Figure 4. General I/S planning approach

A significant piece of the BSP methodology is the formulation of a recommended information architecture. The intent is to develop an implementation strategy that can be built in modules and yet provide justifiable return to the business at each step. These modules or implementable pieces are generally referred to as information systems (I/S). Each could be thought of as the depository or management point for a particular set of the data classes. As the data classes are implemented (assumed through data base technology), it is possible to provide the information needs for various business processes. Each I/S then normally becomes associated with one or more business processes and one or more data classes (see Figure 5). The implementation strategy should obviously be in tune with the business needs. An identified I/S is responsible for the collection and maintenance of its data bases for the entire business. Other, or later implemented, systems can draw on these data bases as well as new ones that they will create and maintain.

Most of the identified I/S areas will normally be in support of processes that are operational in nature. That is, the decision areas inherent in the processes will relate to the "operational control" level of planning and control. Thus these I/S's are sometimes referred to as *operational control systems*. The managed data is also at an operational level and is typified by much detail and quantity. A *management control system*, on the other hand (Figure 5), is in support of a "management control" category of processes. Its managed data would typically be summarizations of the internally generated operational data (dotted lines in the figure) plus any other data (competitive data, for example) as necessary to support the processes. By having "business plan" data in the data base, the executive could effect control decisions by comparing actual versus plan for his area of responsibility.

In summary, there are a number of basic concepts and philosophies relative to information systems upon which the BSP methodology is based. The methodology itself should be considered flexible in nature. That is, certain steps and techniques could be altered in order to adapt to specific situations without detriment to the final outcome. However, these basic concepts themselves should be considered inviolate. They, in effect, *are* BSP.

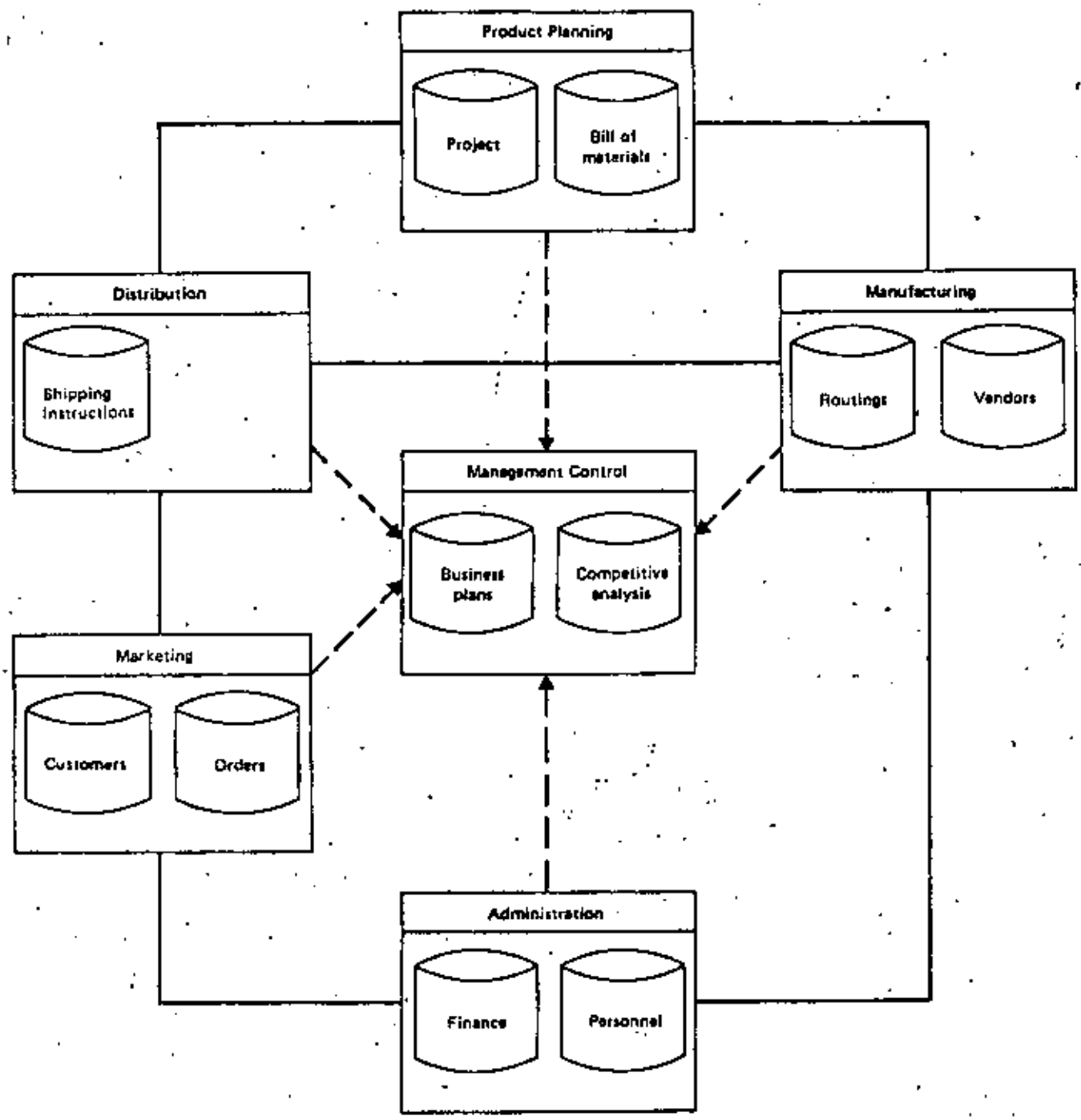


Figure 5. Information architecture example

Chapter 2. Overview of the BSP Study Approach

The keys to success in planning, developing, and implementing an information architecture that effectively supports the business goals are:

- Top-down planning with bottom-up implementation
- Managing data as a corporate resource
- Orientation around business processes
- Use of a proven, comprehensive methodology

Chapter 1 examined the first three at some length. This chapter gives an overview of a proven methodology for the BSP study which will be delineated in more detail in the following chapters.

Perspective

Since BSP is part of an overall cycle for providing the business with required information, it should be put into perspective. As Figure 6 shows, there is a major juncture between identification of overall business requirements and the six project phases for implementing an I/S. Requirements are identified for a total business unit and then separated into projects that are undertaken and implemented over time. In addition to information systems projects, there is also a continuing set of projects covering information architecture and information systems management (ISM). Identification of overall business requirements is not reiterative unless a major change occurs within the business unit that would change the basic business processes.

BSP is a methodology for identifying the business requirements. If an entire business were not included in the BSP study, but only a component such as one profit center, additional BSP studies would be appropriate for other business components.

In the past, and in many businesses today, projects are defined to address a functional area of the business without regard to the total requirements of the business unit. BSP can provide overall direction for the total business before projects are undertaken and therefore avoid the fractionalization of data and inconsistencies of systems.

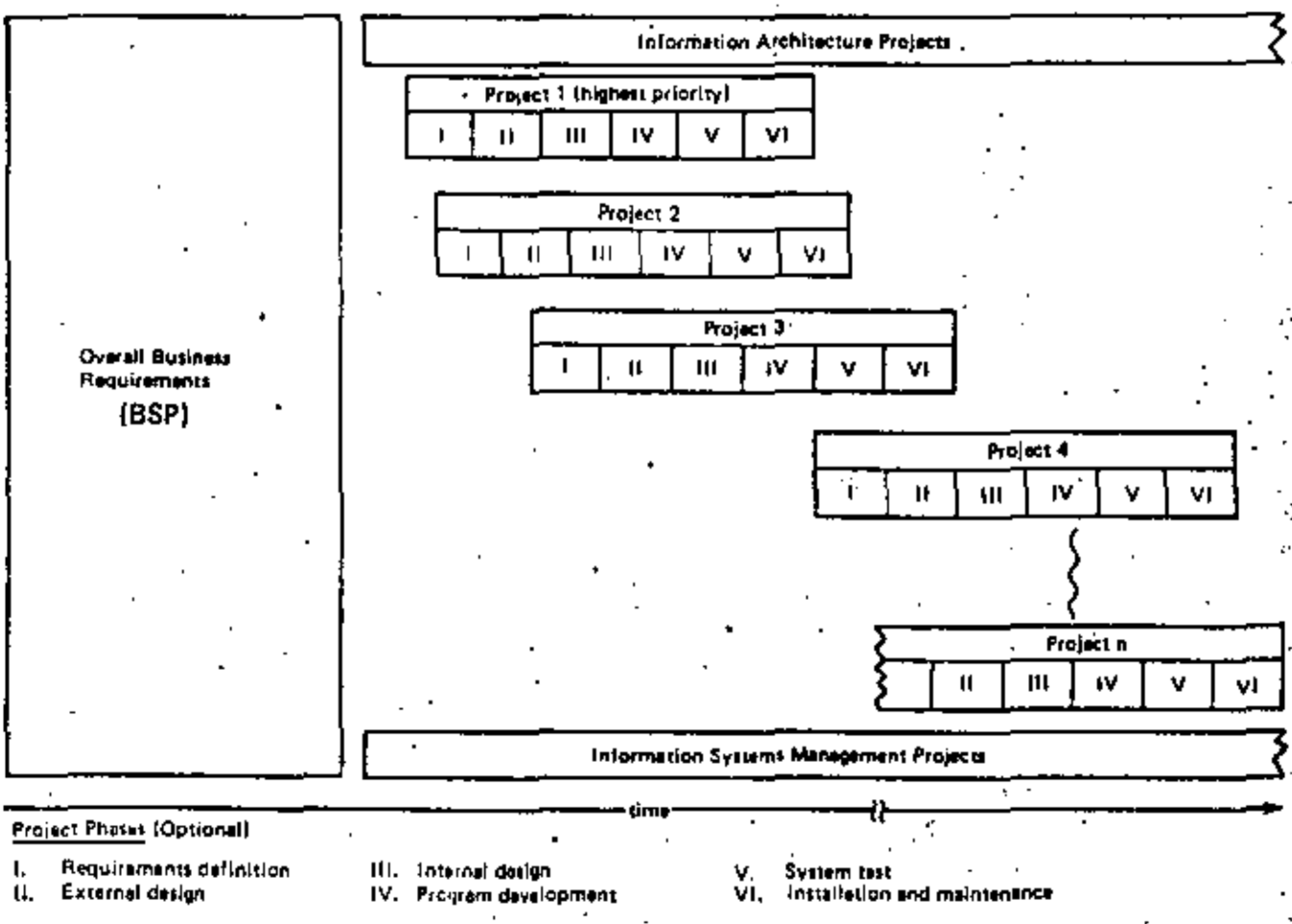
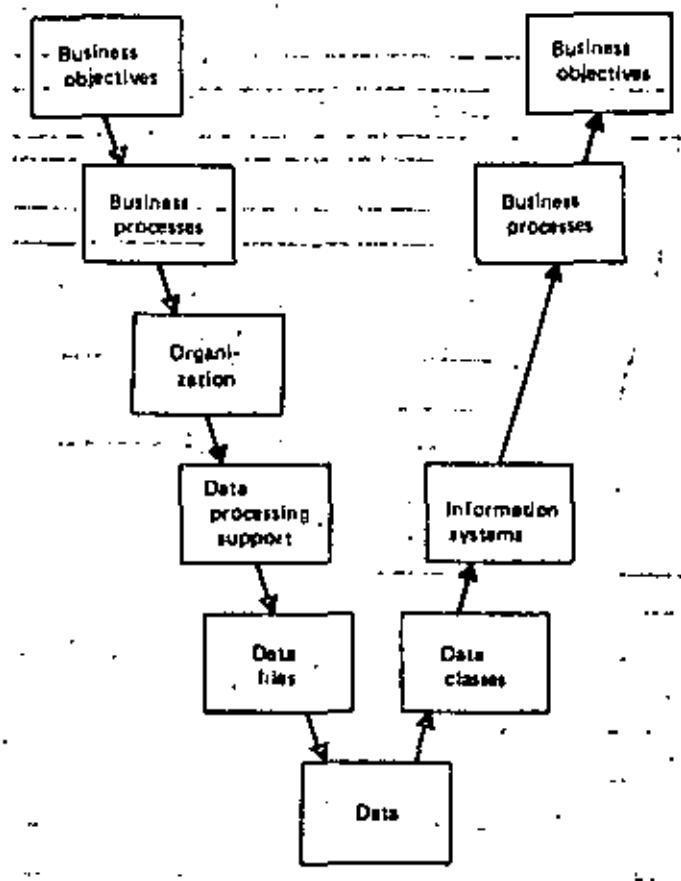


Figure 6. Relationship of BSP to I/S projects

The top-down, bottom-up aspects of the BSP study are reflected in the following:



The study progresses from the very broad world in which the business exists down to the data required to run the business. The data is categorized into data classes that lead to the definition of information systems to support the business processes and business objectives. The study starts with a collection of facts about the business that are usually available in documented form throughout the organization. These facts are organized, abstracted, and analyzed by the study team and enhanced by the top executive, who explains the business and adds those points usually not documented. The study progresses to an identification of the major activities and decision processes in the business and then each of the executives is asked to validate and enlarge upon the facts that have been gathered and analyzed. The analysis ends with the consolidation and comparison of the facts from all sources. From this understanding the study follows the normal path of findings and conclusions, recommendations, action plan, and executive presentation for concurrence and support.

Major Activities

As Figure 7 indicates, there are two major activities that precede a BSP study and eleven in the study itself.

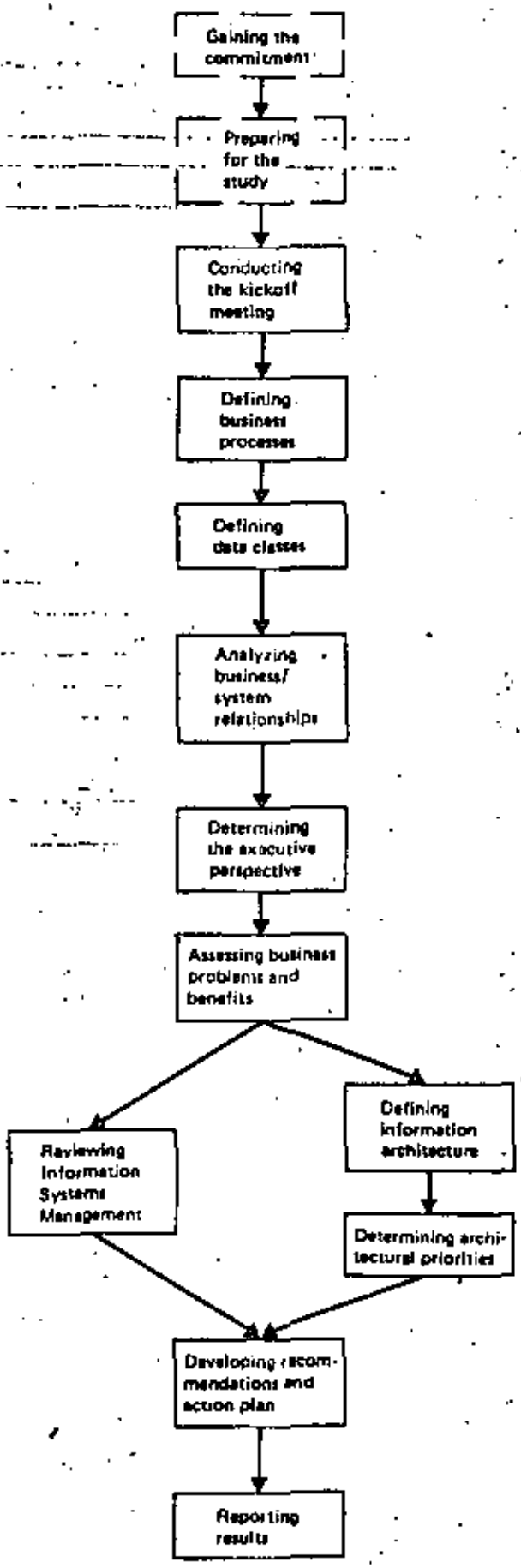


Figure 7. Flow of the BSP study

While these activities may be carried out in varying degrees, none can be omitted. Figure 7 shows these activities and their most logical arrangement. When one becomes fairly familiar with the BSP approach, however, he can, as appropriate, do part of a given activity and delay the balance. The remainder of this chapter is a brief commentary on these major activities.

Gaining Commitment

A BSP study should not be started unless a top executive sponsor and some other executives are committed to being involved in it. The study must reflect their view of the business, and the success of the study depends upon their providing the business understanding and information requirements to the team. Most of the input will come directly or indirectly from these executives.

Since approval of study recommendations commits the company for several years to a certain direction in the use of its data processing resources, it is important at the outset to get agreement on the scope and objectives of the study and on its expected deliverables, so as to minimize future misunderstandings.

The most important action following commitment regards selection of the team leader, an executive who will work full time in the study and direct team activities. He sees that contact with other executives is on the proper level and that input from them is interpreted correctly. A letter from the sponsor to all participating executives sets the tone and signifies commitment.

Preparing for the Study

Considerable saving of time, avoidance of frustation, and higher quality of output can be gained by proper study preparation. All executive participants and the team need to know what will be done, why, and what is expected of them. Proper education and orientation will provide the best input from the executives and the best use of it by the team.

Interviewees are selected as soon as possible so as to allow for their orientation, scheduling of interviews, and the providing of information to the study team. For maximum efficiency on the part of the team in working together full time during the study, information on the company and on data processing support is gathered before the study kickoff.

A control room is established so that the team may work together, display relevant material on the walls, and conduct interviews.

The major output should be a study control book containing: a study work plan; a schedule of interviews and a schedule of checkpoint reviews with the sponsor; an outline of the final report from the BSP study; and business and information systems data, analyzed and charted, and ready for the kickoff period. This activity should end with a review by the BSP study sponsor.

Conducting the Kickoff Meeting

The BSP study itself and the full-time participation of the team members starts with the kickoff meeting, which consists of three presentations. First, the executive sponsor reiterates the objectives, expected outputs, and perspective of the study with relation to other company activities and objectives.

The next presentation is concerned with the main purpose of the kickoff, which is to provide that each team member is conversant with the information that has been gathered and to discuss those facts that are not part of the information supplied. To accomplish this the team leader "walks through" the business facts that have been gathered and makes subjective comments and additions on facts that cannot be readily documented – politics and sensitive issues, and changes planned and in process. He should also cover the decision process, how the organization functions, key people, major problems, the user's view of DP support, and the image of the DP department.

The third presentation is made by the information systems director or one of his managers, who gives the team a view of data processing analogous to what was presented for the business. He should also cover project status and project control, history of major data processing projects started in the last two years, major current activities, planned changes, and major problems.

These three presentations, added to the facts that have been gathered and made readily available to the team, should give the team an overall understanding of the business and the present and planned data processing support.

Defining Business Processes

No other activity during the study can be quite as overwhelming or as important as the identifying of the business processes. Since these processes form the basis for executive interviews, the information architecture, problem analysis, data class identification, and various follow-on activities, everyone on the team must acquire an understanding of all the processes, and they can do so by assisting full time in their identification and in the writing of their descriptions. The major output from this step will be a list of all the processes, a description of each, and the identification of those that are key to the success of the business.

Defining Data Classes

The defining of data classes is the grouping of data into logically related categories. This classification, and its

subsequent modification during the follow-on projects, helps the business develop data bases over time with a minimum of redundancy and in a manner that allows systems to be added without a major revision to the data bases. Since data must be recognized as a corporate resource, it deserves the attention recommended here.

When the data classes have been identified, they are related to the business processes in order to define the information architecture, and they are related to present files to assist in the development of a migration plan.

Analyzing Business/Systems Relationships

The main purpose of this activity is to show how data processing currently supports the business in order to develop recommendations for future action. The currently existing organizations, business processes, information systems (applications), and data files are analyzed to identify voids and redundancies, help clarify responsibilities, and further the understanding of the business processes.

The main analysis tool is the matrix, and various matrices are developed using combinations of the four elements. The key matrix for the executive interviews is the organization/process matrix, and its intercepts denote the decision makers, major and minor organizational responsibility for a process, and current data processing support.

This activity will not only prepare the team for discussion with executives but will also help them determine requirements for information support.

Determining the Executive Perspective

This activity is an integral part of the top-down approach. Its purpose is to validate the work done by the team, determine the objectives, problems, and information needs and their value, and gain executive rapport and involvement. The executive interviews provide the business understanding necessary for information systems planning.

The major output consists of notes from the interviews, an update of the control room charts, and a new or improved rapport between the executive and the BSP study team.

Assessing Business Problems

Some of the business problems were supplied as input during the fact-gathering step. These were expanded upon and complemented by the knowledge of the team and finally validated, explained, and added to in the executive interview. The problems must now be analyzed and related to the business processes so as to give guidance to the setting of project priorities and to show clearly that better information will help to solve the problems.

One of the last activities in assessing the business problems is to divide them into those which are information systems support problems and those which are non I/S problems. The non I/S problems will be delineated and given to the executive sponsor to follow up on while the information systems support problems continue to be addressed in the BSP study and by the follow-on activities.

Defining Information Architecture

This activity represents a major movement from an examination of the present to a synthesis of the future. It is here that we sketch future information systems and their accompanying data.

Systems may be viewed as the automated portions of processes. Data bases are the computerized part of the total inventory of data in the business. Information architecture brings order and structure to the systems and the data they create and use. Once it is structured, it allows for step-by-step development to migrate from the applications of today to the information systems of the future.

Because this task involves drawing a blueprint for the future, it deserves the attention of the whole team.

Determining Architectural Priorities

Since a total information architecture cannot be developed and implemented at one time, the team must establish priorities for the development of the systems and data bases. By deciding which of the data bases should be designed and implemented first, the team establishes which subsystems will be defined during the follow-on project. This allows an early implementation for a "pay-as-you-go" foundation and helps establish credibility for the balance of the output from the BSP study.

Prioritization is accomplished by developing a list of projects from the subsystems of the information architecture, then establishing a set of criteria and rating the prospective projects against them. The analysis of the business problems is a major contributor to this process.

Reviewing Information Systems Management (ISM)

The purpose of ISM is to eventually establish a controlled environment in which the information architecture can be developed, implemented, and operated efficiently and effectively. The information systems functions are examined during the BSP study to identify (1) any changes that could be made immediately to enhance success in the follow-on projects, (2) changes

that are necessary to properly manage and implement the high-priority information architecture projects, and (3) major activities that will become projects in the follow-on to the BSP study. Fulfillment of this step is vital to the successful support of the business by the data processing function.

The major inputs are the information systems support problems from the executive interviews, the ISM problems as identified by the I/S director, and the technologies and skill requirements of the first system or systems. Additions and/or changes to the present information systems functions that are required for planning, developing, and implementing are then identified and put in priority sequence. This forms the basis for the development of an action plan for ISM which feeds into the action plan and report from the BSP study.

Developing Recommendations and Action Plan

The purpose of the action plan is to assist management in its decisions regarding the recommended follow-on projects. The follow-on projects will come from the architectural priorities and from the information systems management recommendations. Each of the projects will have been defined as a result of the activities in those two areas. The action plan brings these together to identify specific resources, schedules, and interactions of the projects.

Another major part of the action plan should be the identification of the steps preparatory to starting each of the follow-on activities. These must be examined and sized before start dates can be put on the follow-on projects, so that management can give the direction necessary to move immediately into those preparatory activities.

Reporting Results

The purpose of the final report and executive presentation is to obtain executive management commitment and involvement for implementing recommendations from the BSP study. The format of the report was agreed upon before the study was started, but may have to be modified slightly now, on the basis of the study results. Various parts of the report should be written during the BSP study and finalized at this time.

The report should be prepared as follows:

1. It should include an executive summary.
2. The supporting detail, such as descriptions of the business processes, should be contained in appendices.
3. It should be possible to extract highly confidential or sensitive material easily and still have the balance of the report usable by all people participating in the follow-on activities.

The report provides the basis for the executive presentation and the distribution of the final results to those people designated by the sponsor. After the executive presentation, which is normally given by the team leader, all other relevant material that is not a part of the report should be indexed and filed for ready availability in follow-on projects.

Chapter 3. Gaining the Commitment

The success of a Business Systems Planning study depends heavily upon the commitment of the business to complete all the relevant activities and heed the recommendations of the study team. The steps described in this chapter should be completed and an assessment made of the commitment of all participants, before the final decision is made to move into the resource dependent activities. Certain situations and environments will be considered unsuitable for a BSP study at this stage. If such is the case, the team should be prepared to postpone BSP until the situation becomes more suitable.

In some organizations there may be a problem in obtaining access to executive management to explain the objectives and expected results of a BSP study. If so, the following actions should be considered:

- Arrange an executive visit to another business that has successfully completed a BSP study
- Invite a senior IBM executive to discuss information systems planning with the appropriate executive in your organization
- Conduct an executive briefing session specifically for your executives
- Conduct an executive planning session – a service offered by IBM
- Invite the executive to an executive information session conducted at an IBM Customer Executive Education center

The IBM representative can help set up any of these activities.

Establishing the Study Scope

Normally the executive sponsoring the study (called the *executive sponsor*) will select the scope of the organization to be studied. This business unit must be defined so that all participants can recognize the boundaries within which their activities will be concentrated.

In general, the selected segment should extend to the upper levels of the organization, include more than one major function, and be of significant importance to the organization in that it contributes a major portion of the revenues, profits and/or services.

Businesses whose activities are participated in by multiple functional units tend to gain more from a BSP study than those that are more simply structured.

Most companies performing a BSP study choose a major operating division, a group of divisions, or the entire organization (see Figure 8, options 1-3 respectively). In each case it should be possible to define

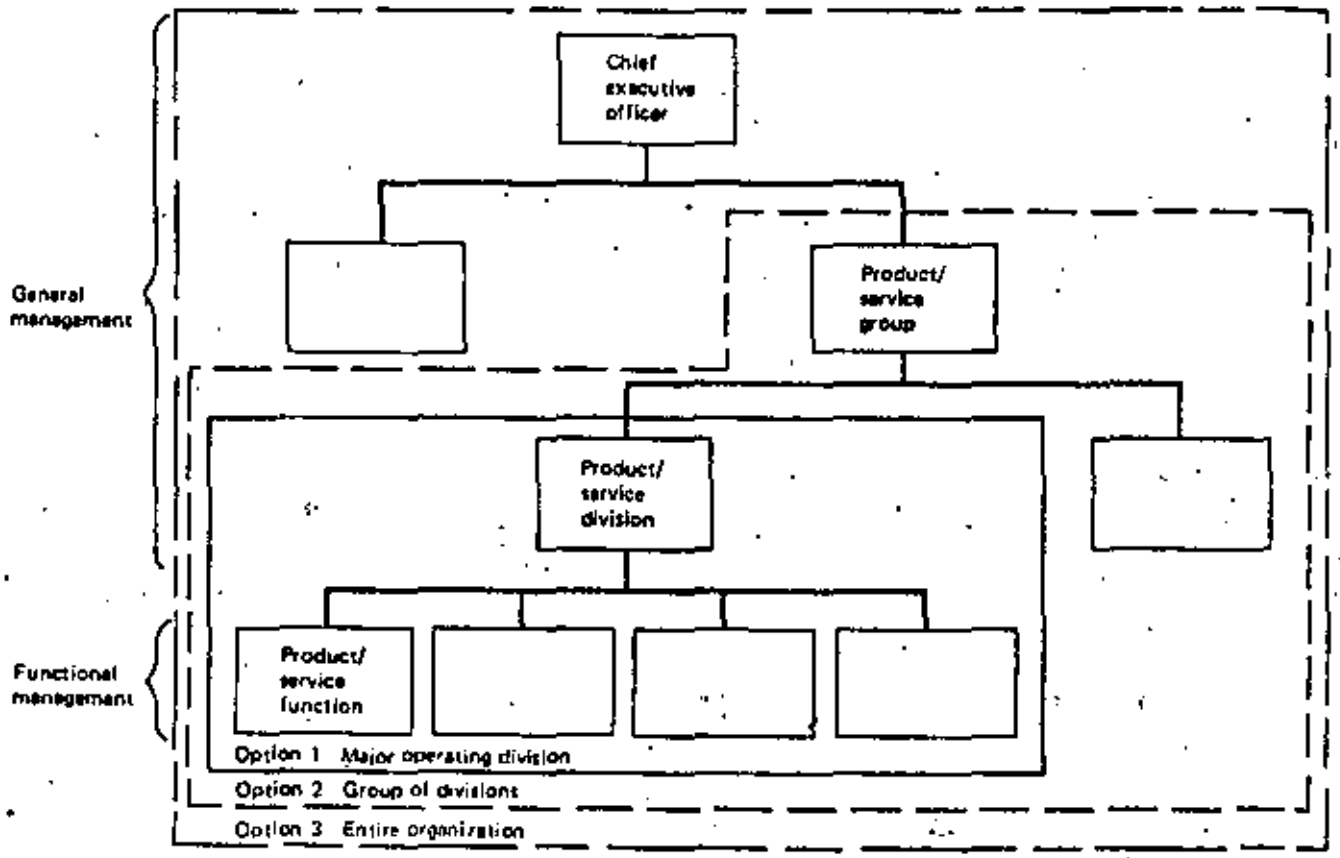


Figure 8. BSP organizational business unit options

boundaries for the study scope, including the measurements for assessing effectiveness.

Setting Study Objectives

Study objectives should now be defined. Good objectives will be capable of being stated clearly and concisely. They will have criteria that can be applied to measure the degree to which they have been met. In addition, they will be applicable to the specific requirements of the organization to be studied. Finally, they will be capable of being satisfied by the BSP study team working within the defined study scope.

Specifically, a study team's objectives may be to:

- Provide a formal, objective method for management to establish information systems priorities without regard for provincial interests
- Provide for the development of information systems that have a long life, protecting the systems investment
- Provide that the data processing resources are managed for the most efficient and effective support of the business goals
- Increase executive confidence that high-return information systems will be produced
- Improve relationship between the information systems department and the users by providing systems responsive to the users' requirements and priorities
- Identify data as a corporate resource that should be planned, managed, and controlled, in order to be used effectively by everyone

Developing Business Reasons for the Study

There are several factors that may make a given business more difficult to do a BSP study for than others:

- Major reorganization or transformation of control taking place
- Geographical diversity, necessitating significant travel time
- Well over 20 executives who must be interviewed
- Multiple independent sub-organizations with autonomous decision making processes and with unique lines of products/services serving diverse markets

These factors should be reflected in a brief report (two or three pages) containing:

- Statement of the study scope
- Study objectives
- Positive contributions of a BSP study
- Obstacles to successful completion
- Recommendation either to proceed with BSP or postpone it

Current participants should present this report to an executive of the organization and gain agreement on all issues. Assuming that the report recommends proceeding with BSP, the meeting should result in a commitment to:

- Appoint a study team leader
- Resource the study team as appropriate
- Have open communication of future plans and current data among all team members
- Write a study announcement letter to executives of the organization of the form shown in Appendix A.

Resourcing the Study Team

Characteristics

Since the study team is responsible for determining the information needs of the entire business and recommending the nature of its data processing operations for years to come, the selection of team members is very important.

The people chosen should:

- Have several years' experience within the organization, a sound knowledge of their own area, and an appreciation of the rest of the business
- Be able to understand and deal analytically with problems
- Be willing to commit to conclusions and recommendations that will have a far-reaching effect on the organization
- Be perceived by other management within the organization as competent and responsible businessmen whose opinions will carry considerable weight

In short, the best people for the study team are those who are already in high demand and difficult to make available.

Organization and Responsibilities

A possible organization structure for the study team is depicted by Figure 9. While minor variations are possible, the following fundamental responsibilities need to be assigned:

- Executive Sponsor
  - Visibly endorse and support the study effort.
  - Review progress and findings during the study.
  - Receive the final report and make the approval decision.
- Team Leader
  - See that the study is successfully completed.
  - Provide overall direction, make key business judgments, and act as liaison with other executives of the organization.
  - Direct the study effort on a day-to-day basis and organize all necessary administration for the team.

- Team Members
  - Participate in at least some, if not all, interviews.
  - Analyze the data collected and executive perspectives expressed during the study.
  - Draw conclusions and perform report writing tasks.
- Team Secretary
  - Do typing, filing and general secretarial services.

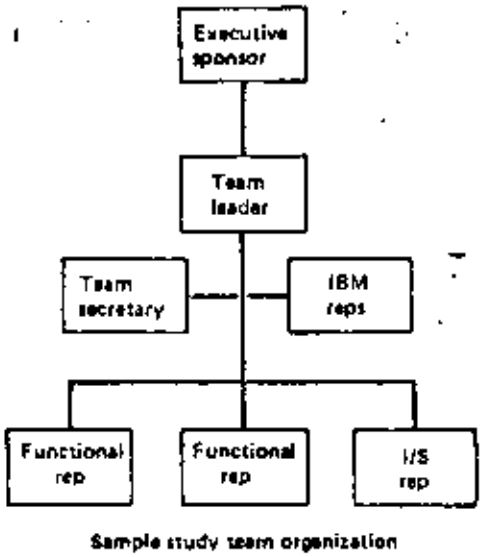


Figure 9. Sample study team organization

Considerations for Full-Time or Part-Time Study

To compound the problem of team selection, a BSP study is intensive in nature, and is best performed on a full-time basis. While part-time studies may appear to be easier to resource and may be successful, they are often characterized by deadline and continuity problems. Before embarking on a part-time study, assurance should be gained that:

- The team will be very well managed. That is, the time, effort and skills required to tightly control the work schedule must be available and committed.
- Momentum will not be lost as a result of inactivity during the study.
- Astute use will be made of the administrative support available to effectively utilize the time when the team is not in session to prepare and type drafts, update charts, etc. One full-time member may be helpful.

Manpower Requirements

If the intent and methodology of BSP are reasonably followed, and the study team is full time, the resources required should fall within the following limits:

- Number of team members 4-7
- Number of weeks 6-8

These wide variations can be explained by significant variations in organizational types and sizes, as well as study objectives.

Selection of Team Leader

The success of the study depends largely on the person selected by the executive sponsor to lead the team. The team leader will be knowledgeable in business matters and have a broad perspective of the business. With first-hand knowledge of how the various departments of the business interact, and where detailed information about the operation of the business can be obtained, the leader can save the team valuable time.

Activities of the team leader include:

- Conducting the study team orientation session
- Conducting a kickoff meeting for team members at the outset of the formal activities of the BSP study
- Setting and confirming executive session schedules
- Managing the logistics of the study, including day-to-day administration
- Providing guidance and business perspective
- Providing resource material
- Maintaining the schedule
- Presenting the study report to management at the completion of the study

Sponsor's Letter

The team leader is now in a position to compose a study announcement letter to be signed by the chief executive officer and distributed to the executives who control the major functions within the scope of the study. The letters should cover:

- Objectives of the program
- Potential value to the organization
- Need for functional executives to be involved
- Need for candor and cooperation of executives

A sample announcement letter is included in Appendix A.

Chapter 4. Preparing for the Study

Using the criteria established in Chapter 3, potential team members are identified. Final selection is normally performed by the team leader and the executive sponsor. The information systems representative on the study team will provide continuity for the activities that will follow the BSP study. Each team member must be thoroughly briefed on the magnitude and importance of the study and on the executive sponsor commitment to BSP.

A preparation plan will be prepared by the team leader with assistance from IBM personnel. The major tasks are:

- Briefing the study team
- Gathering data on the business and information systems
- Educating team members
- Developing the study work plan
- Defining the study report and ancillary output
- Creating an executive interview list and schedule
- Briefing participating executives
- Obtaining a suitable control room
- Arranging for administrative support
- Reviewing with executive sponsor

These tasks, many of which can be performed concurrently, are described in the remainder of this chapter.

Briefing the Study Team

The team leader, assisted by experienced IBM people, should hold a half- to full-day orientation session to brief the study team on such topics as:

- Overview of BSP. This can include development of the BSP concept, study methods used, output that may be expected, and references to results obtained in other BSP studies.
- Review of activities to date. The team leader presents the scope of the study, states its objectives, and leads a discussion of the business reasons for conducting the BSP studies at this time.
- Study preparation plan and schedule. Generally the team leader will have prepared a draft plan and schedule. After this has been presented, the team, operating as a working committee, should critically review the plan.

The leader should also discuss with team members the allocation of the preparation activities. As a work group, the team should divide the activities among themselves. In general, distribution of the workload among the various team members depends on how much time they can devote and how closely the tasks relate to their respective functions.

Gathering Data

To provide a basis of information the team should establish a reference library of both business and information systems documentation. Data of value will be of two types:

General Business

- Environment
  External – economic, customers, technology, competition, government, suppliers
  Internal – corporate policies, practices, constraints
- Industry position and industry trends
- Planning process and calendar
- Business plan – goals, objectives, strategy, resources, schedules, financial
- Organization charts – position, names, numbers of people, and expected changes
- Key decision makers
- Major business measurements and control
- Major studies, task forces, committees during the previous three years
- Major problem areas
- Products and markets
- Geographic distribution
- Financial statistics
- Sizing statistics (vendors, employees, orders, customers, shipments, purchase orders, etc.)
- Project initiation and funding process

Information Systems

- I/S plan – goals, objectives, strategies, resources, schedules, finances
- Policies and practices
- Organization charts
- Planning process
- Major studies, task forces, committees during previous three years
- Geographical distribution (equipment, terminals, communication facilities)
- Software environment
- Project initiation and funding process
- Systems justification requirements
- Standards/guidelines
- History of major projects started in the last two years
- Present and planned systems and data bases
- Major information systems problems
- Sizing statistics (number of programs, jobs per period, users, files, data elements, and turnover)

Educating the Team

To enable a smooth, coordinated entry to the BSP study, the team members should understand the principles of the BSP methodology. Some understanding of the method of implementation, plus a review of the nature of the output that can be expected, will also be of advantage to the team members.

The team leader, working with IBM personnel, should establish an education plan for the team. Formal IBM education is available and the local IBM office can supply details.

It is recommended that all team members should attend the same education session.

Developing the Study Work Plan

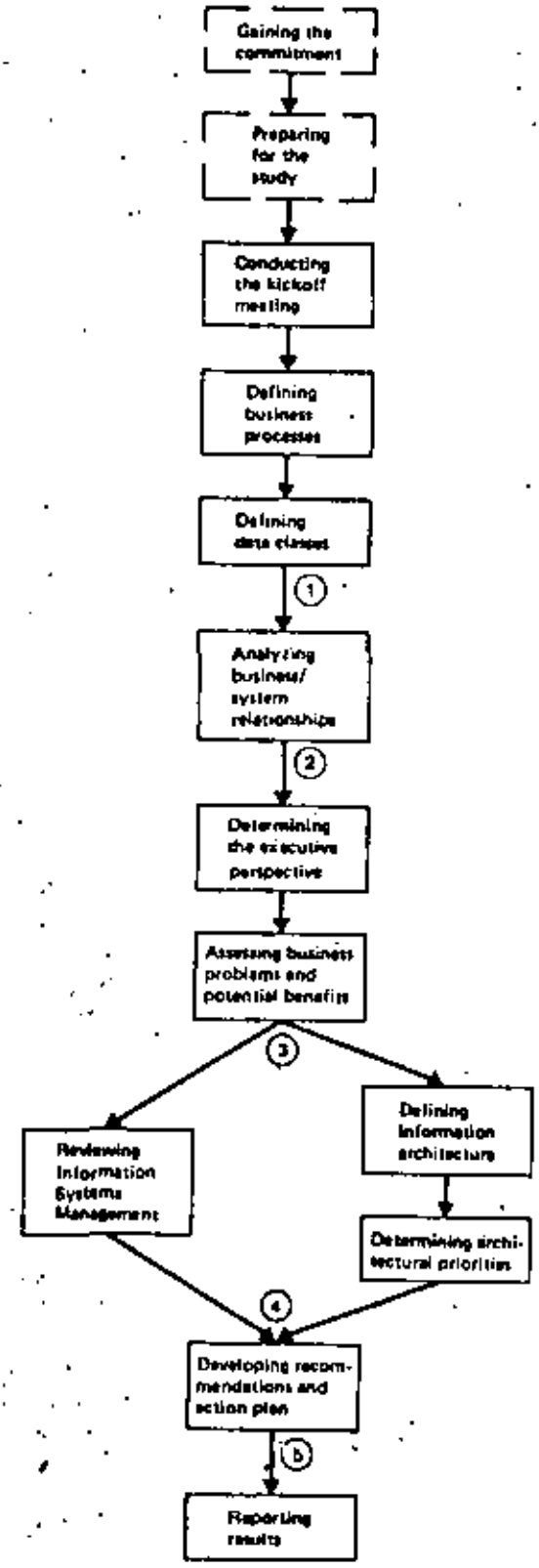
Intensive study activity requires appropriate planning and assignment of tasks. A work plan will document all the known tasks and indicate major resource allocation required to complete the study. The team, working under the guidance of the team leader, should develop a study work plan in keeping with the scope and objectives previously established.

There are some study control points that should be scheduled in the work plan (e.g., see Figure 10). The team should consider whether additional control points should be established in each particular study.

When the activity and project control checkpoints have been established, they should be combined into a single action plan. This plan can be represented on a Gantt chart similar to that shown in Figure 11. This plan is developed under the control of the team leader, and agreed upon by the team members.

Control Point	Participants	Activities
①	Study team	Review study to date, documentation standards, and team understanding of results. Confirm resource allocations for the next stage.
②	Executive sponsor	Results to date. Update the study plan. Review the executive interview objectives.
③	Executive sponsor	Report on both qualitative and quantitative results of the executive interviews. Present and validate the assessment of business problems and benefits. Update the study plan.
④	Study team	Team agreement on all major issues. Review all supporting documentation to be completed. Confirm resource allocation. Update the plan.
⑤	Executive sponsor	Review all major findings and recommendations. Demonstrate an understanding of the business and its requirements. Gain executive sponsor's agreement that the team is qualified to present the recommendations.

Figure 10. BSP study control points



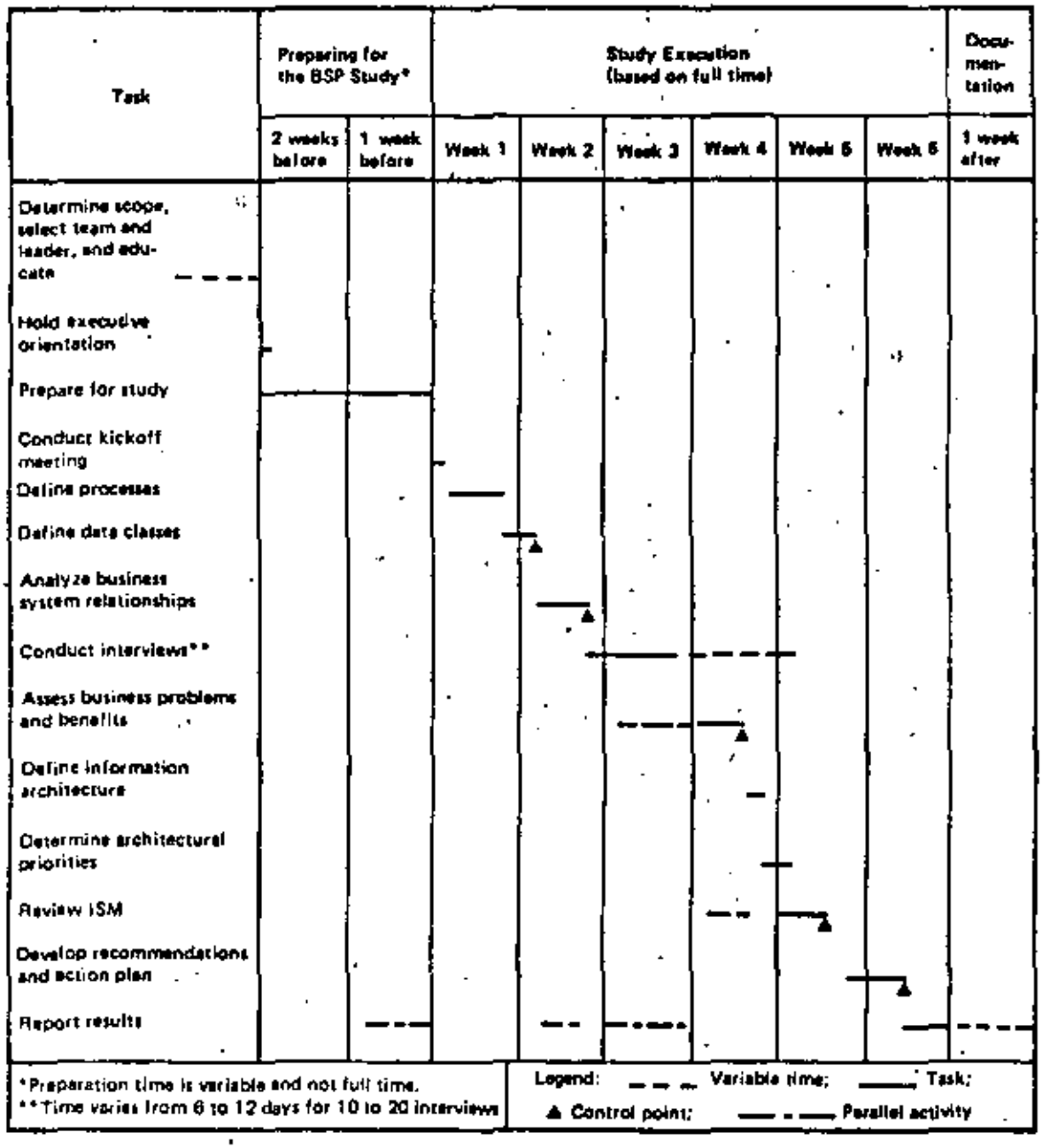


Figure 11. Work plan for the BSP study

Establishing a BSP Study Control File

The control file should be established to contain all material pertaining to the study. Because of the nature and number of the documents involved, they should be stored in a lockable filing cabinet. Suggested contents include:

- Key correspondence
- Statements of objectives and scope
- Project plan
- Schedules
- Data gathered during preparation
- Documentation of completed analyses
- Executive interview notes
- Executive interview summaries
- Project control review meeting notes
- Recommendations
- Report and presentation

Defining the Structure of the Final Study Report and Ancillary Output

The outline (and table of contents) should be established as early as possible. Team members should be assigned specific responsibility for sections of the report. During the study period, it is the assigned member's duty to coordinate and deliver their section.

A typical Study Report will contain the following topics:

- Executive summary
- Purpose, scope and objectives
- Method of study
- Business perspective
- I/S perspective
- Findings and conclusions
- Recommendations
- Action plan for follow-on projects
- Appendices as required

Because of the nature of a BSP study, documentation created during the study may prove valuable to functions within the business. While the data and analyses are current, the team should endeavor to publish these documents as ancillary output to the formal report.

Creating Executive Interview List and Schedule

A preliminary list of executives to be interviewed should be prepared as early as possible. The executives selected will normally be within the top three levels of the company and be responsible for the major functions included within the study scope statement. Executives with cross-company responsibilities should also be considered. Since the need is for a perspective rather than specific detailed information, the ideal candidate is the senior executive with discrete functional responsibility, e.g., director of finance, manager of production planning, director of marketing operations.

The tentative executive list and schedule should be prepared by the team leader. Some other names may be added to the list during the study as the team gains knowledge of the company's processes. When creating the schedule, the team leader should consider all factors that may inhibit the full participation of the executives -- vacation, important business meetings, public holidays, etc.

The list of proposed executive participants should be reviewed and approved by the executive sponsor. Additions or deletions should be made and the support of the executive sponsor reemphasized.

To enable the executives to prepare for the interview, an invitation letter should be sent at least one week before the interview. The letter should also confirm the time and place for the interview. Reference to the study announcement letter previously signed by the chief executive officer should emphasize the importance of executive participation in the study (see Appendix B). Either within the body of the invitation letter or as an attachment, a list of topics for discussion during the interview should be included. Both the executive and the team members should use this topical outline to make the discussions more fruitful. Possible topics include:

- Responsibilities
- Objectives and plans to achieve required results
- Business problems and value of solving them
- Opportunities for improvement
- Major measurements and controls
- Identification of the key business areas
- Anticipated changes (products, organization, etc.)
- Evaluation of current I/S support
- Requirements for future I/S support
- Value of the required I/S support

Briefing Interviewees

The selected executives should attend a briefing session prior to the commencement of the executive perspective segment of the BSP study. During this preparation phase, the executive briefing session should be prepared, presenters selected and exhibits prepared, so that the session may be delivered at the appropriate time during the study activities.

Major topics which should be included in the briefing are:

- Executive level overview of BSP
- Study scope
- Objectives of the study
- Business reasons for conducting the study
- Review of the expected output
- Executive involvement – why, and what is expected of the executive
- Schedule for executive interviews

Obtaining and Equipping Control Room

Effectiveness of the BSP study depends on having a workroom assigned for its duration. This room is used both as the work location for the team and for executive discussion session away from the executive's office. This approach has proved very effective in allowing executives to review charts and diagrams on display and to concentrate on discussions without interruption.

The study room should be:

- Located away from traffic and noise
- Large enough to accommodate the study team plus one or two other people
- Equipped with a large table, blackboard, flipchart stand, and lockable storage or filing cabinet
- Able to accommodate the posting of charts on the walls
- Securable to protect sensitive information
- Equipped with a telephone only if the telephone can be disconnected during executive interviews

Arranging for Administrative Support

Typing and general administrative support will be required by the study team. This support should be estimated and organized before the major study activities are undertaken. Administrative activities will be concentrated toward the latter part of the study and will include executive discussion summaries, work plans, status reports, results of analyses, charts, final report, confirmation of interviews, and general secretarial support.

Reviewing with Executive Sponsor

Completion of the study preparation is indicated by a successful review meeting with the executive sponsor. Any issues that may have arisen during the preparation should have been resolved by this time.

Items to be reviewed with executive sponsor are:

- General results of the preparation
- Study work plan
- Definition of the study output
- Approval process
- Executive selection and discussion schedule
- Letter of invitation to be sent to selected executives

Everything is now prepared for the start of the BSP study. Experience has shown that if all the items discussed in Chapters 3 and 4 have been completed, the study results should be of high quality.

Chapter 5. Conducting the Kickoff Meeting

Conducting the kickoff meeting is the first major activity in the formal execution of a BSP study. This, and the next five major activities (see Chapters 6-10) are all aimed at understanding the business requirements and data processing support as it exists today as well as business requirements for the future. Since this understanding is vital to the success of the study, all these activities must be carried out conscientiously.

Sponsor's Opening

The kickoff meeting should be attended by all study team members. The executive sponsor opens the meeting, emphasizes the business reasons for initiating the study, and explains the significance of this high-priority program to the company. The sponsor may also reiterate the objectives and scope of the study and expected output, and offer personal commitment and support in the interest of seeing a successful completion to the study.

Review of Study Work Plan

The study team leader reviews the work plan and the Gantt chart or calendar developed during the study preparation phase. The work plan is used as a point of departure for discussing and finalizing:

- Major events, major outputs, and schedules
- Individual assignments
- Project control

This review can also serve as an opportunity to discuss administrative details, such as secretarial support, handling of phone messages, working hours, office security, expense charge numbers, etc.

Business Review

The team leader should be prepared to present, in summary form, the business related information gathered during the study preparation phase. This information may be new to most team members and should serve to broaden the business perspective of the team as a whole.

Presentation emphasis should be given to business goals, objectives and strategies; business problems; organizations; geography; environment; and to information relative to understanding the major resources of the business – cash, people, facilities, materials, and products/services.

Some of the material presented will be continually focused upon throughout the study. This selected information should be posted in the control room. The team leader may also elect to distribute hard-copy handouts to each team member for their own personal review and/or research. In any case, all presentation material, including detail backup, should remain in the control room for continued reference. Much of it will be placed in the Study Control File.

Information Systems Review

The director of I/S, or the team member representing I/S, should present, in summary form, the material gathered during study preparation. This will enable the study team to understand how I/S is currently supporting the business and planning to support it in the future.

The major projects installed or being developed should be discussed with the study team. The discussion should include whether these projects have met or will meet their objectives.

Another area that should be discussed is how data processing interfaces with its users, the responsibility it has in the development and justification of applications, and who has responsibility for data within the company.

A copy of the presentation should be made available to the study team members, as it will help them later in the study – especially during the interviewing and when the team members are developing their conclusions and recommendations.

The following is an outline for presentation of the I/S function:

- Mission and objectives
- History and background of data processing
- Organizations
- Overview of major application systems installed and planned
- Relationship with users
- Project approval process
- Key problems and challenges

Chapter 6. Defining Business Processes

For purposes of the BSP study, business processes are defined as *groups of logically related decisions and activities required to manage the resources of the business.* They are studied and identified without regard to the organization responsible for them. The reason for defining the processes is that doing so will provide or lead to:

- Information systems with a large degree of independence from organization changes
- A comprehensive understanding of how the business accomplishes its overall missions and objectives
- A basis for separating the strategic planning and management control processes from operational control processes
- A basis for defining required information architecture, determining its scope, defining it so as to make it modular, and setting priorities for its development
- A basis for defining *key* data requirements

Lists of processes, along with some sample descriptions from both the public and private sectors, may be found in the appendices.

Prerequisites to Defining Processes

The following points are particularly relevant to success:

- All members must be present and participate throughout the exercise and there should be general agreement on the expected outputs before the exercise begins.
- Note-takers should be designated right from the beginning so that decisions and definitions of processes will not be forgotten or misunderstood later.
- Information gathered before the start of the BSP must describe and size the products and resources, and clearly state or provide a flow diagram of the strategic planning and management control procedures and schedules.
- The team must understand the concept of resources and their life cycle.

The Product and Resource Life Cycle

The manager's job is to manage the resources within his realm of responsibilities to most efficiently and effectively support the goals of the total business. By identifying the decision processes that he goes through and the activities that he performs in managing the resources, it is possible to gain a comprehensive understanding of all the business processes involved. The products/services may be defined as key resources and occupy a major role in defining the business processes.

A four-stage life cycle of the product/service and of each of the supporting resources is used to logically identify and group the processes. The life cycle normally used is:

- Stage 1 – Requirements, planning, measurement and control
- Stage 2 – Acquisition or implementation
- Stage 3 – Stewardship
- Stage 4 – Retirement or disposition

The following paragraphs refer to these stages as requirements, acquisition, stewardship, and retirement.

A description of each of the life cycle stages follows:

1. *Requirements* – activities that determine how much of the product or resource is required, the plan for getting it, and measurement and control against the plan.
2. *Acquisition* – activities performed to develop a product or service or to get the resources that are going to be used in its development. In manufacturing this would include procurement and fabrication; in personnel, the hiring or transfer of people; in education, the development of a curriculum and the enrollment of students.
3. *Stewardship* – activities to form, refine, modify or maintain the supporting resources and to store or track the product/service. In the insurance industry this could be policy maintenance, premium notices and dividend statements; in the distribution industry it could include inventory control and warehousing.
4. *Retirement* – those activities and decisions that terminate the responsibility of an organization for a product or service or signal the end use of a resource. This might include the selling of space on an airline, the discharge or retiring of an employee, the scrapping or selling of a capital asset, or the removal of a service by a government agency.

The life cycle serves as a vehicle for structured, logical, comprehensive identification of the processes by the team.

Basic Steps in Defining Processes

An overview of process identification is provided by Figure 12, which shows the three main sources for the identification of the business processes: planning and control; product/service; and supporting resources. Taking the latter two through their life cycle provides a definition of their respective business processes. Because strategic planning and some of management control are not solely product or resource oriented, they must be considered as a separate source to provide that all processes of the business are identified.

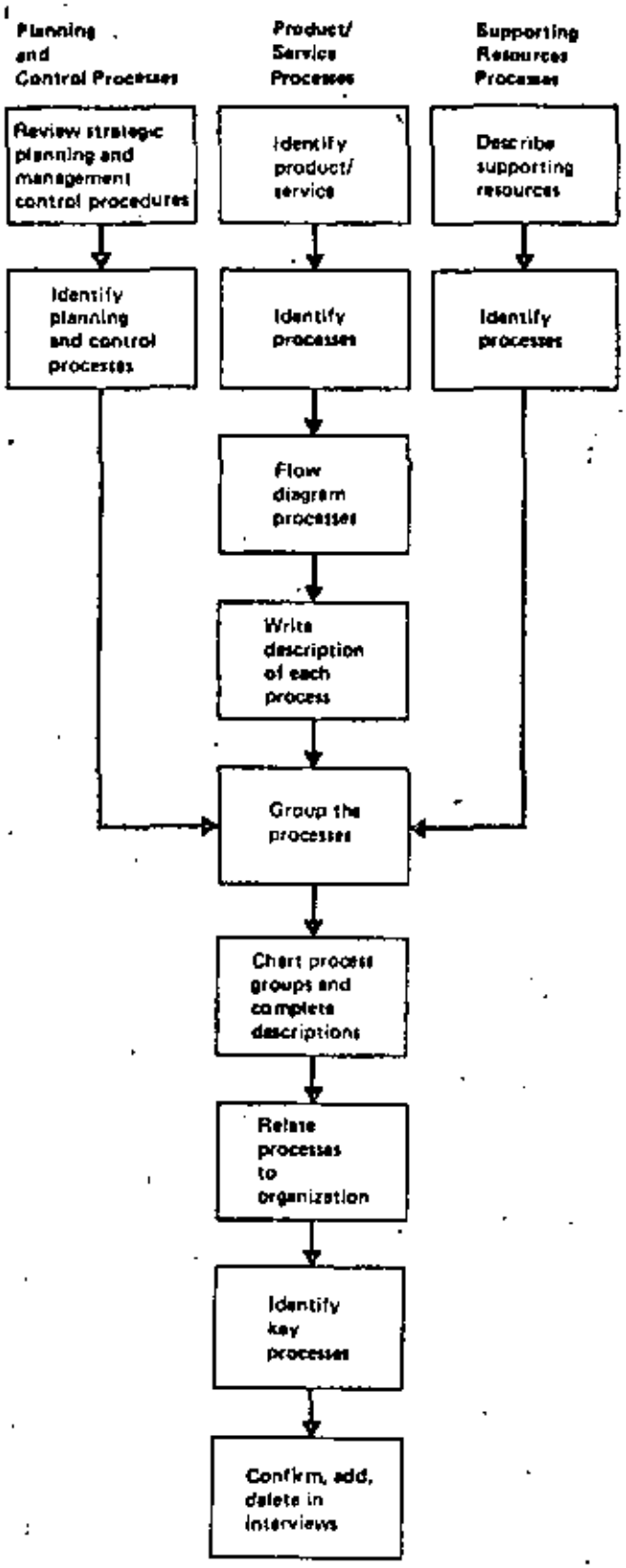


Figure 12. Definition of business processes

After the product and each of the supporting resources have been taken through their respective life cycles and the processes identified and subsequently grouped, it is no longer necessary to retain the process/stage relationship for process defining, but it will be helpful later in identifying data classes. Since the life cycle is an artificial means for directing the team's thinking, very little time should be spent in deciding in which stage a given activity appears. The recording of the activity is more important than its position.

The measurement and control processes identified under the requirements stage will contain both management control and operational control activities. If the processes associated with strategic planning and management control are identified first, it will sharpen the distinction between management control and operational control and reduce some of the redundancy in the management control processes. The following descriptions of steps correspond to the flow presented in Figure 12.

Planning and Control Processes

Define the Processes of Strategic Planning and Management Control

With the preparatory work that was done in collecting the information on planning and possible samples of the organization's plans, it should not be difficult to identify the processes involved. They will normally be grouped into strategic planning and management control. Strategic planning may be referred to as the long-range plan, the seven-year plan, or the development plan. Management control may be referred to as the operating plan, the management plan, the resource plan, and sometimes the contract plan. In some companies the budget may serve as a major tool for management planning and control. Examples of planning and control processes are shown in the following table. Further examples are contained in Appendix D.

Strategic Planning	Management Control
Economic forecasting	Market/product forecasting
Organization planning	Working capital planning
Policy development	Staff level planning
Divestiture/acquisition analysis	Operational planning
Assumptions management	Budgeting
Goals development	Measurement and review
Product line modeling	

For a detailed discussion of the theory of strategic planning and management control the reader may wish to refer to Robert N. Anthony's book entitled "Planning and Control Systems: A Framework for Analysis."

Product/Service Processes

Identify the Product/Service of the Organization
For purposes of this exercise it is assumed that there is one major product group or that the products are managed through similar business processes. In the case of the public sector and some service organizations it will help to go back to the goals to best specify the product or service. For example, one might think that the product of a health insurance organization would be its policies, but an examination of its goals might show the products to be (1) service to the subscriber and (2) payment to the provider. In like manner, an examination of the goals might keep one from assuming that the curriculum is the major product of a university.

If there are two or more groups or families of products and services, they are discussed under "Expansions and Variations" at the end of this chapter.

Identify the Processes in Each Stage of the Life Cycle of the Product/Service
The general approach in identifying the processes is to start with the requirements stage and work through the succeeding stages. Care should be exercised in getting consistency in the level of the processes identified in each of the stages. Although there is no prescribed number of processes appearing in each of the stages, most studies result in only 20-60 processes for the whole business. The team should tend to identify more processes than necessary, with the idea of grouping them later as necessary.

With the total team working together it is sometimes helpful to label each of four charts with the four stages and to complete them simultaneously as each process is brought into discussion. This eliminates the development of notes for the inclusion of certain items in other stages as they are worked upon. There is no secret formula for this identifying of business processes, so one should not be surprised if the first attempt results in many more processes than are needed and also a great inconsistency in the levels. Do not let this deter you; it will be corrected during the grouping of processes.

It is very important that every team member participate at all times, since business processes are the basis for practically everything that follows in the ASP study. Getting started is far more important than any theoretical discussions on the definition of a process group or a process, logic, theory, or other diversions.

The following example might result from performing this analysis with a manufacturer of electrical and electronic components.

Requirements	Acquisition	Stewardship	Retirement
Marketing planning	Engineering design and development	Inventory control	Selling
Marketing research	Product specifications	Receiving	Order servicing
Forecasting	Engineering records	Quality control	Shipping
Pricing	Production scheduling	Packing and storing	Fleet management
Material requirements	Production operations		
Capacity planning	Purchasing		

Make a General Flowchart of the Product/Service Processes
A flowchart of the product/service processes (see Figure 13) serves several purposes:

1. It helps in providing that all the processes have been identified.
2. It helps in determining if the team really understands the business processes associated with the product/service.
3. It serves as a model for the subsequent definition of the information architecture.
4. It helps in identifying the processes involved in managing the supporting resources.

There should be a box on the flowchart for each of the processes. If other boxes seem to be required, the definition of the processes should be reexamined. The flowchart will result from the identification of the above processes and is not intended to show the strategic planning and management control processes.

Write a Description of Each Process
A member of the team should have been making notes on the decisions and activities associated with each of the processes as they were defined by the group. Although this is a laborious task, it is absolutely mandatory if the process descriptions are going to reflect the knowledge of the team and the amount of time that has been spent in defining them. The note-taking job can be rotated through the team and not be a burden on any one individual. The descriptions may be either in list form or verbal.

Completing these descriptions before moving into the next step will help bring out the similarities and differences between these processes and those subse-

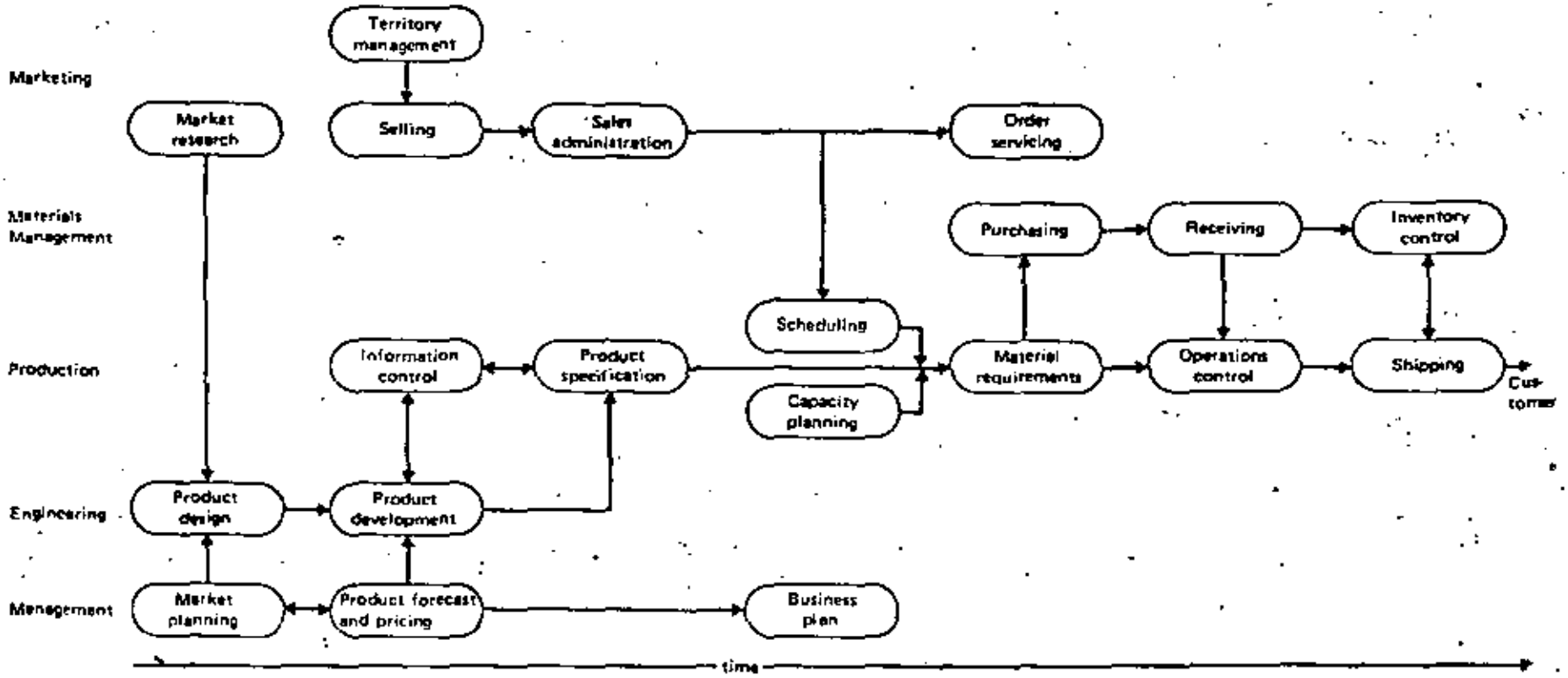


Figure 13. Example of the flow of a product/service through a business

quently identified. One of the advantages of rotating the note-taking among the different members is that the task of further defining the processes can now be assigned to most of the team members so that it may be accomplished before a description of the supporting resources is started. The final revision of these descriptions will probably be included in the appendix to the final report upon completion of the BSP study.

The following are examples of process descriptions (further descriptions may be found in Appendix E):

Production planning – the activity of planning for and coordinating materials, men and machines in order to produce the finished products needed to meet forecasted requirements

- *Capacities and capabilities planning* – the process of balancing production need with ability
- *Scheduling* – the scheduling of labor and material needs to meet production and shipping requirements. Also, the scheduling and balancing of the product mix
- *Material requirements* – the calculation of raw material needs to meet a schedule, recognizing optimum inventory levels, economic order quantities, substitutions, etc.
- *Costing* – establishing of standard raw material costs and product costs on the basis of these and other manufacturing and administrative cost factors

Purchasing – the activities of acquiring materials, machinery and supplies of specified quality on a timely basis at the best price

- *Supplier evaluation and selection* – the searching for, evaluation of, and selection of suppliers who meet requirements for materials, packaging, machinery, equipment and delivery at competitive prices (includes internal search)
- *Order placement and follow-up* – the actual placing of purchase orders with approved suppliers for quantities of raw materials as specified (may include raw material routing) by production planning, equipment acquisitions approved by management, etc. (includes release of customer-purchased materials)
- *Receipts and inspection* – the receiving or returning of purchased materials, machinery, and supplies, verifying quantity and quality, and documenting the activities

Supporting Resource Processes

Describe the Supporting Resources

A resource is best described as *that which a business consumes or uses in meeting its goals.* There are four basic resources that we deal with:

- Materials
- Facilities
- Money
- Personnel

While the use of only these four resources should be adequate, the inclusion of some ancillary resources might make it easier for the team to get a comprehensive list of processes. Selection can be made from the following list:

- Marketplace (general public, potential customers, and customers)
- Vendors
- Documented knowledge (designs, specifications, text material, procedures, patents)
- Company image (in the stock market, to the general public, customers, vendors, employees)

A thorough examination of the activities and decisions associated with these resources should result in the identification of the rest of the operational control processes in which the business engages and which were not brought out in the product/services processes.

The team should review the facts on resources that were gathered and prepared during the preparation for the BSP study. Before starting to identify the processes, it is well to place a chart on the wall with a short description or list of the components of the resource being considered.

Identify the Processes in Each Stage of the Life Cycle of Each of the Supporting Resources

The resources are taken one at a time through the four stages in the same manner as the product/service. The following chart is an example of the processes that might be associated with the resources.

	Life Cycle Stage			
Resource	Requirements	Acquisition	Stewardship	Retirement
Money	Financial planning Cost control	Capital acquisition Receivables	Portfolio management Banking General accounting	Accounts payable
Personnel	Manpower planning Salary administration	Recruiting Transfers	Compensation and benefits Career development	Termination Retirement
Material	Requirements generation	Purchasing Receiving	Inventory control	Order control Movements
Facilities	Capital equipment planning	Equipment purchase Buildings management	Machine maintenance Furniture and fixtures	Equipment disposition

Consolidation and Analysis

Group the Processes

The processes have been identified from the three sources: strategic planning and management control; major product/service; and supporting resources. The grouping should take place along the following lines:

- Reduce inconsistencies in level. In the above examples of the processes, the process of manufacturing and the process of accounts payable appear at the same level. This should be corrected.
- Combine the processes where commonalities occur. As an example, purchasing may have been identified as a process in the acquisition stage, not only for the product but also for materials and for facilities. Assuming the processes are similar, these should be combined into one process. As the processes were grouped during the requirements stage, there may have appeared processes entitled manpower loading, develop workflow plan, and schedule supply materials. These three might be grouped with the management control process of operational planning.

A workable maximum is 60 processes. There will normally be 4-12 process groups. Some will be associated with the resources from which they were derived.

Marketing	**Facilities Management**
Planning	Workflow layout
Research	Maintenance
Forecasting	Equipment performance
Sales Operations	**Administration**
Territory management	General accounting
Selling	Cost planning
Administration	Budget accounting
Order servicing	
Engineering	**Finance**
Design and development	Financial planning
Product specification maintenance	Capital acquisition
Information control	Funds management
Production	**Human Resources**
Scheduling	Personnel planning
Capacity planning	Recruiting/development
Material requirements	Compensation
Operations	
Materials Management	**Management**
Purchasing	Business planning
Receiving	Organization analysis
Inventory control	Review and control
Shipping	Risk management

Figure 14. Sample list of processes by process group

The product/service may result in two or more groups, and planning will account for one or two groups. Figure 14 is an example of grouping. Other examples are shown in Appendix C.

Chart the Process Groups and Complete Descriptions

Each process group and its processes should be listed on a chart and displayed for the whole team to view. The team should agree on the placement of each process under the process group, as well as the name of each, and should have a good understanding of all of them. The list should be divided among the team members and the writing of the descriptions should be completed in line with the regrouping. It is a good idea to have the descriptions typed so that each team may have them for quick reference during subsequent activities such as executive interviewing. It is also advisable to save the charts for subsequent activities.

Relate the Business Processes to the Organization

Once the business processes are agreed upon and described, they can be related to the organizational structure of the business to help the study team identify any additional people that should be interviewed and to further clarify their understanding of the business processes. Some teams may wish to complete this matrix before completing the process descriptions. Relating the organization to the processes also helps the team determine the information needed from the interviews, such as verification of executive involvement in the processes.

To relate the business processes to the organizational structure of the business, the team develops an organization/process matrix. Essentially, this is a graphic representation of one aspect of the management system of the organization because it illustrates who makes the decision in each of the processes.

The organization/process matrix is one of several matrices developed as part of the BSP methodology. Matrices are used because they provide a convenient method to analyze relationships. They also provide a concise way to convey findings to management.

To prepare the organization/process matrix, the study team uses the business processes already identified. Using their knowledge and perspective of the business and the organization charts from the study preparation phase, the team members identify the organizational entities involved in the processes.

Since the BSP study is intended to provide a broad overview of the business, not every organizational entity is identified. Furthermore, common similar organizations can sometimes be represented as one organizational unit. For example, 100 sales offices may be listed as a single unit. Where feasible, plants and laboratories should also be grouped into units. One

unit representation is also generally appropriate when organizations of different scopes are doing the same job. For example, a financial planning organization may have three departments, each focusing on separate divisions. The mission of all three is still financial planning, and they can be represented by one organizational unit. Figure 15 illustrates the organization/process matrix.

Once the processes and organizational entities are arranged on the matrix, the study team completes the matrix by indicating the degree to which each organizational entity is involved in the processes. The following symbols are used in Figure 15 to indicate the degree of involvement:

- ☒ Major responsibility and decision maker
- ☒ Major involvement in the process
- ☑ Some involvement in the process

When making this notation on the matrix, leave room for the posting of systems numbers, as explained in Chapter 8.

Such indicators do not describe the actual responsibilities of each of the organizational units but serve only as a guide to assigned responsibility for and involvement in a process. Some businesses have found this matrix to be valuable after the study as an index to a management system manual, in which they develop responsibility and activity statements for each of the organization/process intersects.

The organization/process matrix helps the study team validate the list of individuals to be interviewed and determine the questions to be asked of the individuals responsible for the processes. To authenticate the matrix, each person interviewed by the team should be asked to confirm or correct the portion of the matrix showing his responsibility or involvement.

The organization/process matrix sometimes will show overlapping responsibility and decision-making authority for a process or a lack of decision-making responsibility for a process where it normally would be appropriate. Such potential problem areas are clarified later during the executive interviews.

Identify Processes Key to Business Success

This step will later aid in selecting the area of the business to be studied in more detail (architecture priority), sizing the importance of the problems identified, and identifying items to be stressed in the executive interviews. The strategic planning and management control processes will be some of the outstanding processes. One method of doing this identification is to rank the objectives in order of their importance and then determine which of the processes are most critical in reaching each objective. The process/organization matrix will also contribute to this selection.

Outputs and Their Uses

The output to be expected from defining the business processes consists of:

1. A list of process groups and their processes
2. A typed description of each of the processes
3. A list of the processes key to the success of the business or designated as such on the charts of process groups and processes
4. Product/services flowchart
5. Team understanding of how the whole business operates and is managed and controlled

As mentioned earlier, business processes serve as the base for most of the activities that follow. The ultimate use of the business processes is to identify opportunities and requirements for the use of information systems to support the business. That is also one of the basic goals of the BSP study.

The team members should at all times keep in mind that they probably will not all be working on the follow-on projects after the BSP study. The study should be regarded as an independent operation and the processes defined in such a way that any experienced person can read the output and understand the work that was done in the study. The follow-on projects will start with the output from the study and further define the processes, the problems associated with each of them, and the information requirements.

Expansions and Variations in Approach

Although following the above procedure should result in well-defined processes and supporting material, other considerations and variations are worth mentioning. The more salient of these are covered in the remainder of this chapter.

Diversity in Business, Product, or Service

If the various divisions or parts of the company or the product/services are too diversified to have common processes, a grouping will be necessary before process identification is begun. When all the processes are identified, it is still highly advantageous to look for processes that might have common information requirements.

Outside Assistance

Even though the team members were selected with the aim of having their combined knowledge cover the entire business unit, there may be voids in that coverage. If so, the team should get outside help where necessary for a thorough understanding of the processes. Care should be taken, however, to limit the time of any outsider invited in and to keep that person at the level at which the team has been working on other processes.

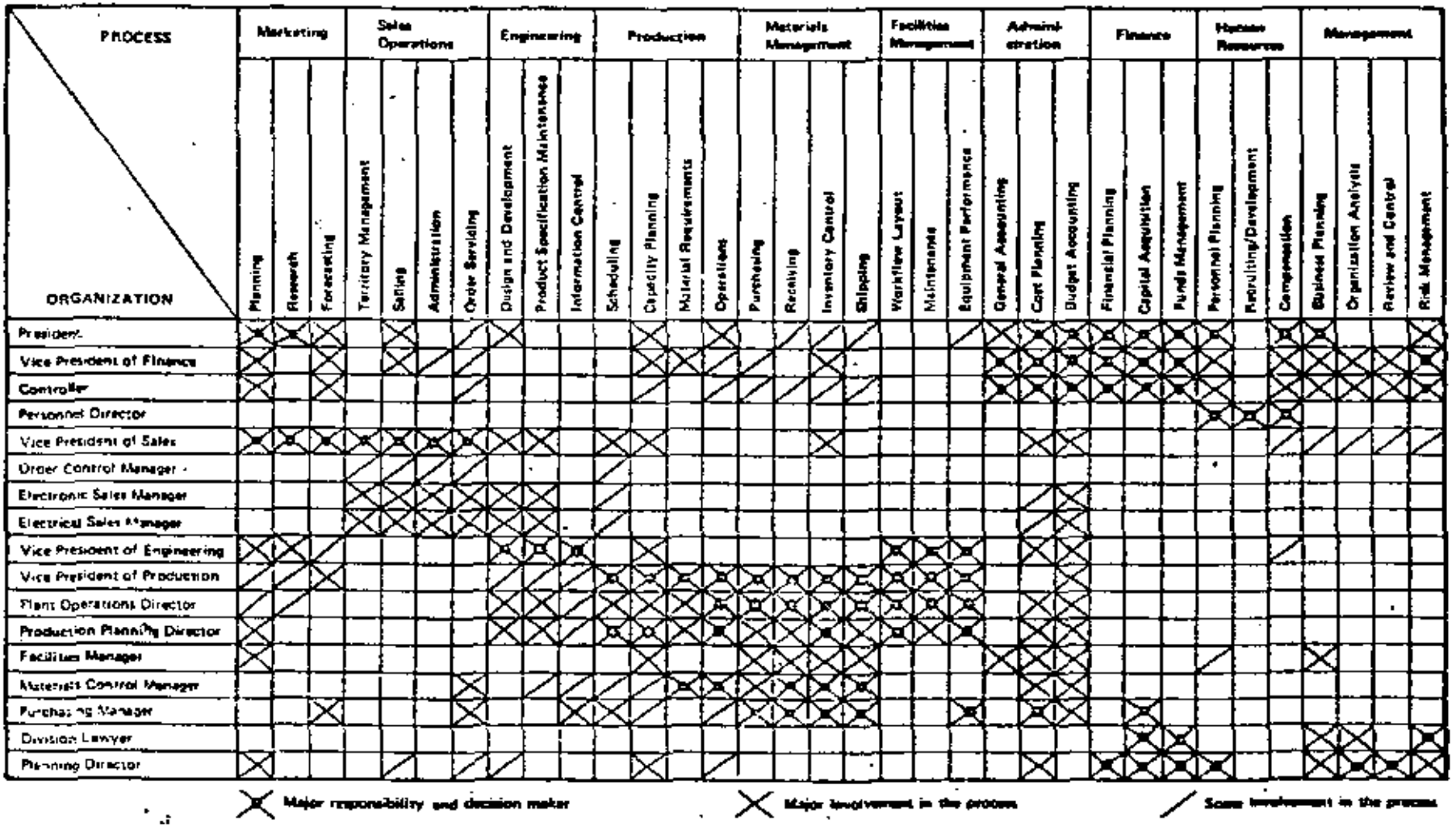


Figure 15. Organization/process matrix

Alternate Methods in Identifying Business Processes

Predefined checklists of processes may be used for the business being studied. These may be found in the appendix, other BSP studies, trade publications, or IBM industry publications such as the Consumer Based Information System (CBIS) or the Communications Oriented Production Information Control System (COPICS). If this method is used, care should be exercised so that the processes are well understood by all the team members and do indeed represent the activities and decisions of the business being studied.

Generic Process Model

An alternate approach in identifying business processes is to use a simple model of the business such as shown in Figure 16.

The model may be expanded to fit the business. For instance, "demand" may become "merchandising" and "selling," while "supply" might break into "product development" and "manufacturing." The following further explains the model:

1. *Supply* includes processes associated with producing the product and obtaining the resources necessary to provide the products or services of the business. These processes normally relate to external interfaces such as suppliers or vendors.
2. *Demand* includes processes associated with making the product or service available and relates directly to external interfaces such as customers or clients.
3. *Requirements* includes processes associated with determining and defining the products or services of the business. These processes normally involve the marketplace and other environmental factors.
4. *Administration* includes processes associated with the overall accountability of the business. These normally involve reporting activities within the organization, such as those directly associated with administration as well as those associated with finance, human resources, and facilities management.
5. *Management* includes processes that tie together the other four aspects of the business. This aspect encompasses the planning, control, and measurement activities of the business.

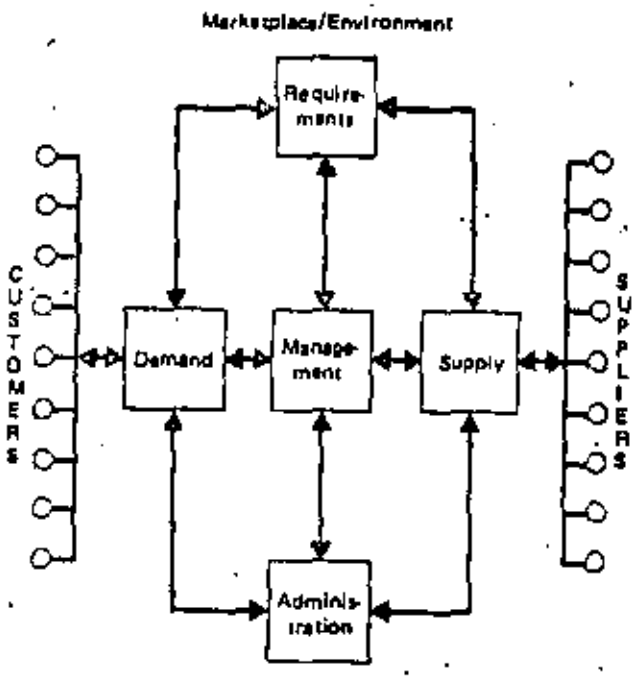


Figure 16. Business process matrix

Chapter 7. Defining Data Classes

A data class is a category of logically related information. Examples of a data class would be customer, vendor, product, order, inventory, etc. Since the purpose of information systems planning is to aid in managing the data resource, the data must be identified. Defining the data classes is one approach to identifying the data to be managed. It also provides valuable assistance in reducing the possibility that data bases developed for early systems projects will require major rework to support later systems projects. Once the processes that support the business have been defined, the next step is to identify the data created, controlled, and used by those processes. As an aid to analysis, the data supporting the processes is grouped into data classes.

Two approaches to identifying data class candidates are presented here, one based on relating data to business entities, the other on relating it to business processes. It is suggested that the two approaches both be used and that their results be cross-referenced to arrive at a final list of 30 to 60 data classes. Data sharing is then illustrated by relating the data classes to the processes that create and use them.

Business Entity Approach

The first approach to identifying data classes is to examine categories or types of data maintained by a business. If the life cycle of resources, previously discussed, is represented as in Figure 17, the types of data related to each stage of the life cycle can be defined.

Planning data, which represents objectives or expectations, supports *requirements* activities. *Inventory* data, which maintains the resource status, supports *stewardship* activities. *Transaction* data effects changes to the inventory data caused by *acquisition* or *disposition* activities. At periodic intervals, *summary* data is extracted from the inventory data to provide feedback on how well requirements have been met. If each business entity is examined for each of these types of data, a set of data classes can be identified.

Entities are simply things an organization or enterprise is concerned about and therefore keeps information about, such as customers, products, materials, personnel, etc. Entities usually include, but are not limited to, the resources around which the business processes were defined.

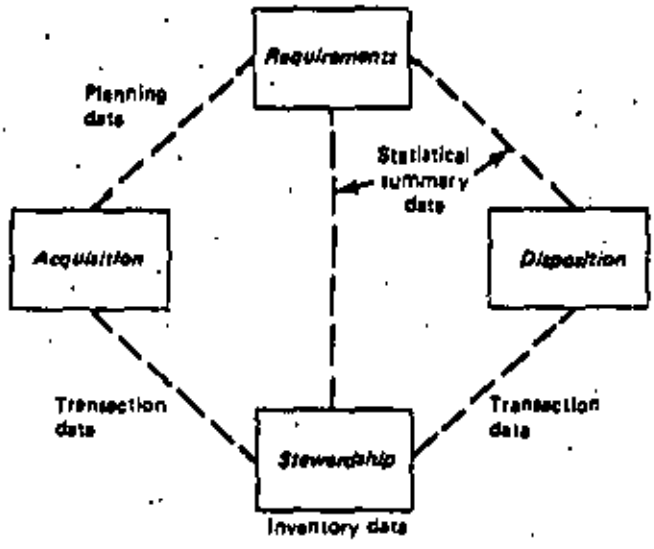


Figure 17. Information life-cycle

Since a data class may contain data relevant to a part of a business entity, an entire business entity, or a group of related business entities, the first step in this approach is to identify and list the business entities. A minimum starting list can be made by referring to the resources around which processes were defined. This list can then be expanded with any other significant entities of the business that have data related to them. Seven to fifteen entities are usually identified.

Next, to identify the data classes related to those entities, a matrix is used, with the entities listed horizontally and the major types of data classes listed vertically. Six types of data class will cover most enterprises' requirements. In Figure 18 the six types are Plans/Models, Statistical/Summary, Inventory, and Transaction. Each entity is examined and the appropriate data names filled in for each data class type under the entity. Inventory data types are usually easiest to identify first, as these are the "master file" kinds of information. Transactions that affect the inventory data can next be identified, followed by the summary and planning types of data.

BUSINESS ENTITIES / DATA CLASS TYPES	Product	Customer	Facilities	Material	Vendor		Cash	Personnel
Plans/ Models	Product plans	Sales territory Market plans	Facility plans Capacity plans	Material requirements Production schedule			Budget	Personnel plans
Statistical/ Summary	Product demand	Sales history	Work in process Equipment utilization	Open requirements	Vendor performance		Financial statistics	Productivity Benefits history
Inventory	Product Finished goods Part master	Customer	Facilities Machine load Routings	Raw Material Cost Bills of material	Vendor		Financial General ledger accounting	Employee Payroll Skills
Transaction	Order	Shipment		Purchase order	Material receipt		Receipts Payments	

Figure 18. Data class/business entity matrix

Business Process Approach

The second approach to identifying data classes uses the previously defined business processes - specifically, what data is created and/or used by each process. This approach involves constructing a series of input-process-output diagrams, one for each of the 20 to 60 processes identified in the business process definition (see Figure 19).

Cross-Referencing and Regrouping

By cross-analyzing Figures 18 and 19 and regrouping on the basis of commonalities or inconsistencies in level, it is possible to compile a manageable list of 30 to 60 data classes.

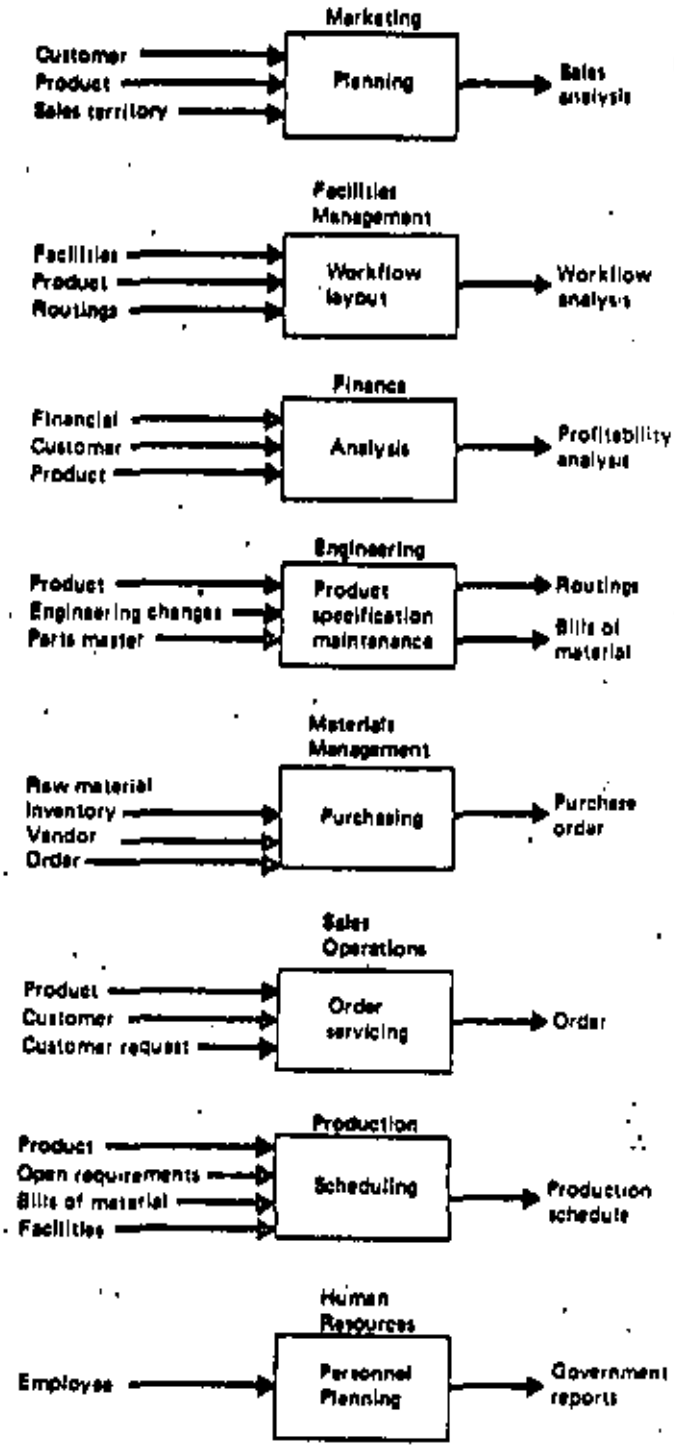


Figure 19. Input-process-output examples

Data Class to Process Analysis

The data classes should then be placed on a matrix opposite the business processes, and the letters C and U should be entered to indicate which processes Create the data and which Use it. Figure 20 shows this plot, with the processes arranged in the life cycle sequence of the key resource. As this matrix shows how data is shared by the processes, it will be used later to develop the information architecture.

PROCESS \ DATA CLASS	Customer	Order	Vendor	Product	Routings	Bill of material	Cost	Parts master	Raw material inventory	Fin. goods inventory	Employee	Sales territory	Financial	Planning	Work in process	Facilities	Open requirements	Machine load
Business planning							C						U	C				
Organization analysis														C				
Review and control													U	C				
Financial planning											U		U	C	U			
Capital acquisition													C					
Research				U								U						
Forecasting	U			U								U		C				
Design and development	U			C		U		C										
Product specification maintenance			U	U		C		C										
Purchasing			C				U											
Receiving			U						U									
Inventory control									C	C					U			
Workflow layout				U	U											C		
Scheduling			U	U											C	U		U
Capacity planning			U		U											U	U	C
Material requirements			U	U		U											C	
Operations					C										U		U	U
Territory management	C	U		U														
Selling	U	U		U								C						
Sales Administration		U										U						
Order servicing	U	C		U														
Shipping		U		U						U								
General accounting	U		U								U		U					
Cost planning		U	U				C											
Budget accounting							U				U		U	U	U			
Personnel planning											C		U					
Recruiting/development											U							
Compensation											U		U					

Figure 20. Data class by process, showing data creation and usage

Chapter 8. Analyzing Business/Systems Relationships

Up to now, the team has been developing a new perspective of the business – learning to look at the business in terms of business processes and the data classes necessary to perform them. Now the team must develop a firm understanding of how data processing currently supports the business, in order to develop recommendations for future action. This section describes the use of data gathered and presented as part of the t/s review as well as the organization, process, and data class information to develop perspectives on:

- t/s support of processes. The organization/process matrix developed in Chapter 6 is annotated to indicate which current systems support the business processes.
- Usage of current data. A system/data file matrix is developed to identify the data files currently in use or planned for use by existing systems.

(A system here is an application or grouping of several applications.)

Each of these matrices is described in more detail in the following pages. Although the matrices provide an overview of the current and planned data processing support of the business, they cannot indicate the extent of support needed or the value of this support to each of the processes. Such information is obtained later during the executive interviews. Used in combination, these matrices are very helpful in providing the team with a broad overview of current systems and data usage.

Understanding I/S Support of Processes

In order to obtain a business-wide picture of existing and planned data processing support, the study team may opt to annotate the organization/process matrix with current systems support (Figure 21). The notations on this matrix identify which organizations involved in the processes are receiving application support. This enables the study team to identify:

- Processes receiving no current systems support
- Processes receiving systems support in some organizational units, but not all
- Possible redundant systems

To create the notations, the team assigns a number to each of the current systems, then posts this number to the appropriate intersections on the organization/process matrix. This may be facilitated by first creating a system/organization matrix (Figure 22) or a system/process matrix (Figure 23).

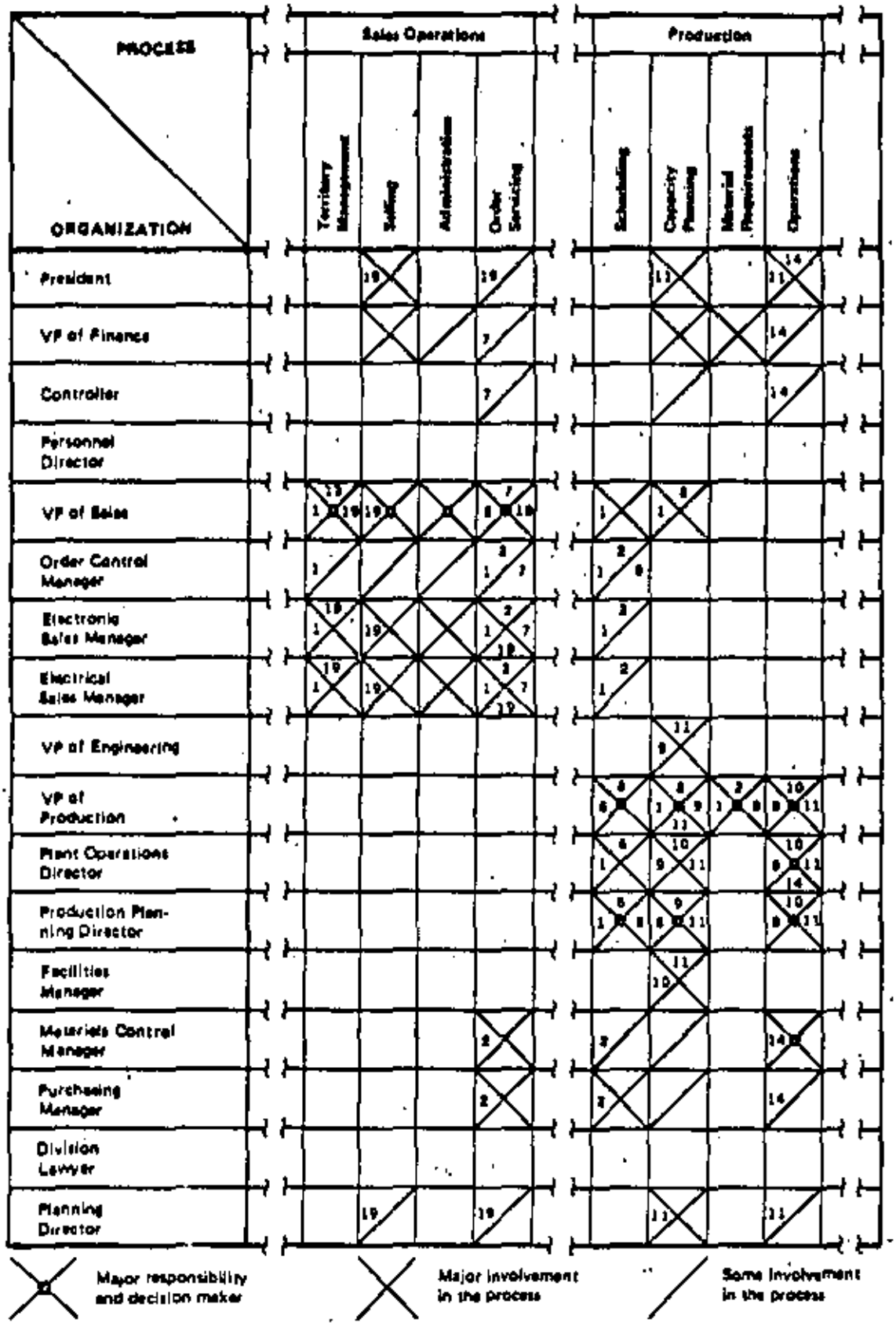
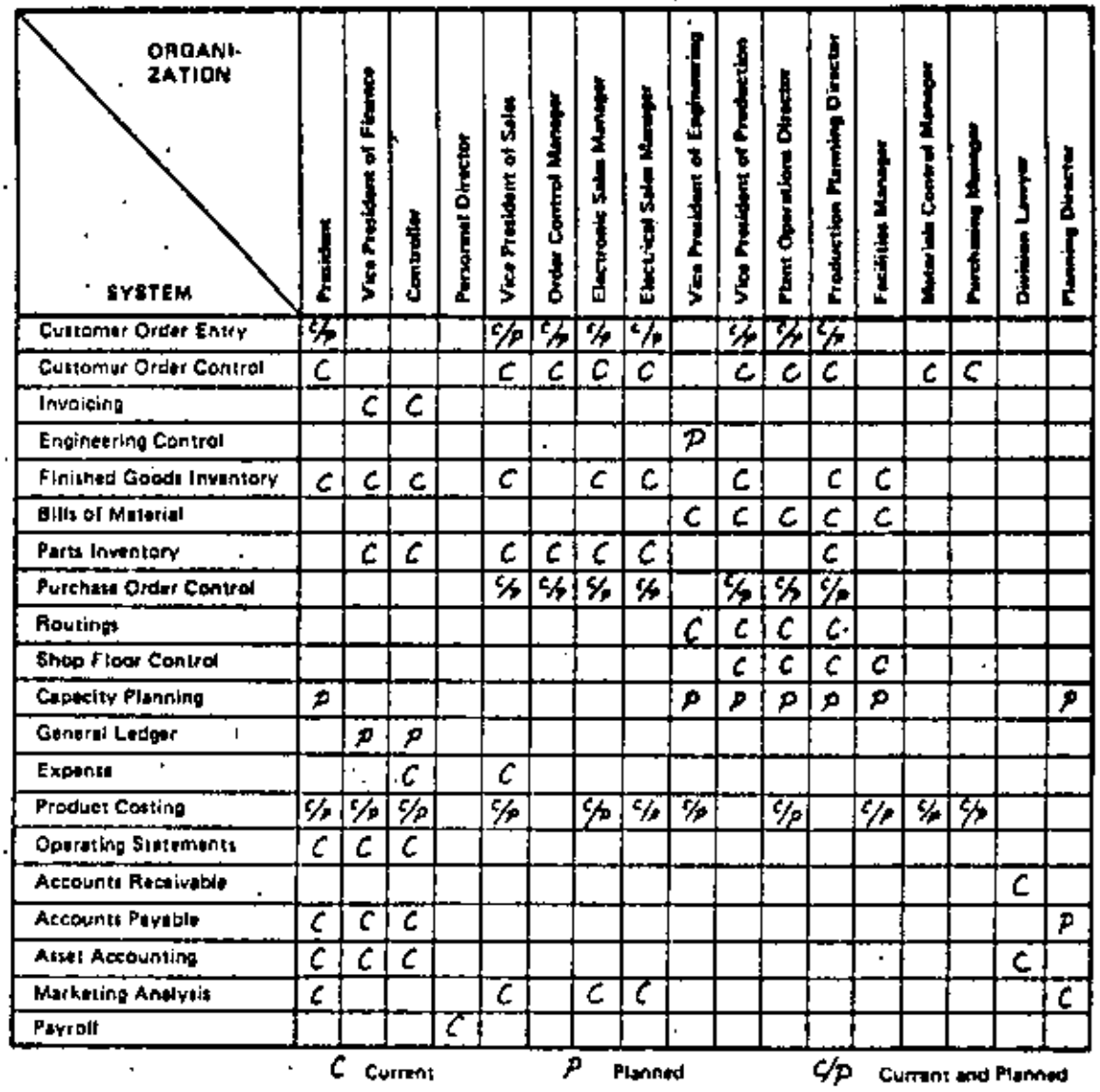


Figure 21. Organization/process matrix, with current systems support shown



ORGANIZATION / SYSTEM	President	Vice President of Finance	Controller	Personnel Director	Vice President of Sales	Order Control Manager	Electronic Sales Manager	Electrical Sales Manager	Vice President of Engineering	Vice President of Production	Plant Operations Director	Production Planning Director	Facilities Manager	Materials Control Manager	Purchasing Manager	Division Lawyer	Planning Director
Customer Order Entry	C/P				C/P	C/P	C/P	C/P		C/P	C/P	C/P					
Customer Order Control	C				C	C	C	C		C	C	C		C	C		
Invoicing		C	C														
Engineering Control									P								
Finished Goods Inventory	C	C	C		C		C	C		C		C	C				
Bills of Material									C	C	C	C	C				
Parts Inventory		C	C		C	C	C	C				C					
Purchase Order Control					C/P	C/P	C/P	C/P		C/P	C/P	C/P					
Routings									C	C	C	C					
Shop Floor Control										C	C	C	C				
Capacity Planning	P								P	P	P	P	P				P
General Ledger		P	P														
Expense			C		C												
Product Costing	C/P	C/P	C/P		C/P		C/P	C/P	C/P		C/P		C/P	C/P	C/P		
Operating Statements	C	C	C														
Accounts Receivable																C	
Accounts Payable	C	C	C														P
Asset Accounting	C	C	C													C	
Marketing Analysis	C				C		C	C									C
Payroll				C													

C Current P Planned C/P Current and Planned

Figure 22. System/organization matrix

PROCESS / SYSTEM	Marketing			Sales Operations				Engineering			Production				Materials Management				Facilities Management			Administration			Finance			Human Resources			Management			
	Planning	Research	Forecasting	Territory Management	Billing	Administration	Order Servicing	Design and Development	Product Specification Maintenance	Information Control	Scheduling	Capacity Planning	Material Requirements	Operations	Purchasing	Receiving	Inventory Control	Shipping	Work Flow Layout	Maintenance	Equipment Performance	General Accounting and Control	Cost Planning	Budget Accounting/Tax Accounting	Financial Planning	Capital Acquisition	Funds Management	Personnel Planning	Recruiting/Development	Compensation	Business Planning	Organization Analysis	Review and Control	Risk Management
Customer Order Entry				C/P			C/P				C/P	C/P	C/P																		C/P			
Customer Order Control							C			C	C		C		C																			
Invoicing																						C			C									
Engineering Control										P																								
Finished Goods Inventory															C	C	C					C	C											
Bills of Material								C	C	C	C				C								C											
Parts Inventory							C								C		C																	
Purchase Order Control											C/P	C/P	C/P		C/P																			
Routings										C		C		C																				
Shop Floor Control										C		C		C			C																	
Capacity Planning												P		P																				
General Ledger																						P				P								
Expense				C																		C												
Product Costing										C/P				C/P			C/P						C/P												
Operating Statements																						C									C	C		C
Accounts Receivable																						C												C
Accounts Payable																						C									C			C
Asset Accounting																								C		C								C
Marketing Analysis				C	C		C																									C		
Payroll																						C		C						C				

C Current P Planned C/P Current and Planned

Figure 23. System/process matrix

Identifying Usage of Current Data

The team needs to understand what data is currently automated and where, that is, which systems utilize which portions of the data. The next matrix developed by the study team, the system/data file matrix (Figure 24), provides this understanding. The systems annotated in Figure 21 form the vertical axis of this matrix, while the data files, grouped by similarity, form the horizontal. An X is placed in each appropriate box to show which data files support which systems.

This matrix sheds more light on how much data is shared by various systems. This, in turn, helps point out the need for a data base approach to provide consistency of data. The information gathered here will also be useful later in developing implementation priorities.

Summary

Some teams find it convenient to use various matrices as communications/presentation tools. Different combinations of two or more (those that share common axes) may be so used.

These charts and matrices provide an overview of the current and presently planned data processing support of the business. Such information should be obtained from the executive interviews regarding problems with current data processing support and additional needs for information. These charts, together with the executive interviews, help the study team determine those areas where emphasis should be applied in developing and enhancing appropriate information systems.

Appendix F contains examples of matrices from other industries.

SYSTEM \ DATA FILE	Customer	Order	Vendor	Product	Routings	Bills of Material	Cost	Parts Master	Raw Material Inventory	Finished Goods Inventory	Employee	Sales Territory	Financial	Planning	Work in Process	Facilities	Open Requirements	Machine Load
Customer Order Entry	X	X	X	X			X			X			X					
Customer Order Control	X	X	X	X	X	X	X			X			X		X			
Invoicing	X	X	X				X						X					
Engineering Control				X	X	X	X	X	X	X							X	
Finished Goods Inventory		X		X				X	X	X			X		X			
Bills of Material		X	X	X	X	X	X	X	X	X			X		X		X	
Parts Inventory			X	X			X	X		X			X		X			
Purchase Order Control		X	X	X				X					X		X			
Routings		X	X	X	X	X		X	X	X					X			X
Shop Floor Control		X	X	X	X	X			X	X					X			X
Capacity Planning		X	X	X	X	X	X	X	X	X			X		X	X		X
General Ledger		X		X			X		X	X	X		X		X			
Expense												X	X					
Product Costing		X		X			X	X					X		X	X	X	
Operating Statements												X	X	X		X		
Accounts Receivable	X	X	X									X	X					
Accounts Payable											X		X			X		
Asset Accounting													X	X		X		
Marketing Analysis	X	X		X									X	X				
Payroll											X							

Figure 24. System/data file matrix

Chapter 9. Determining the Executive Perspective

Interviewing and interview analysis consume more time during the study than any other activity because the executive interviews are the primary source of information for determining the business problem and management's need for support in overcoming their problem and/or supporting new opportunities.

The interviews are not intended to gather data on the specific details and exact form of the information required by the interviewees.

Specifically, the purpose of the interviews is to:

- Validate the data gathered, analyzed, and documented in matrices and charts
- Determine the information needed by individual executives as well as their problems and priorities
- Gain executive rapport and involvement
- Determine management values and quantify benefits to be derived by modifying or adding applications.

The tasks that have to be accomplished in order to conduct the interviews are:

- Confirm the list of executives to be interviewed that was developed during study preparation
- Review the interview schedule to be certain it is practical, and confirm it with the interviewees
- Organize wall charts in control room.
- Review the questions that apply to all interviewees, and develop individualized questions for certain executives, as appropriate
- Determine the role of study team members for each interview and the characteristics of individual interviewees
- Conduct the interview itself
- Prepare a summary of each interview to be approved by the interviewee

This chapter is devoted to a discussion of these tasks.

Confirming the List of Executives To Be Interviewed

During BSP study preparation the team identified the executives that should be interviewed. The team most likely selected managers no more than two levels below the president. Now, aided by the organization/process matrix, the team can confirm its earlier list of interviewees and identify any other executives that should be included.

Two questions can be answered using this approach:

- What organizations are involved in each of the processes?
- Within those organizations, who must be interviewed to determine the problems, objectives, and requirements?

Sometimes it is difficult to assign final responsibility for a process to one specific organizational entity. Several organizations may reach a decision together.

In some cases it is necessary to interview certain managers for political reasons; if they don't have an opportunity to participate in the study, they may oppose any recommendations that result from it. It may also be advisable to interview a particular executive simply because other executives of the same level will be interviewed. This decision normally is made by the executive sponsor.

Reviewing the Interview Schedule

When the list of executives to be interviewed has been updated and confirmed, the study team confirms the interview schedule. Interview preparation time should always be allowed. Experience indicates that an effective interview requires two to four hours, excluding preparation. Therefore, only two interviews should be scheduled for a given day. Normally, 10 to 20 interviews are conducted during the study, although this number varies greatly with the size and requirements of the business.

It is recommended that higher executives be interviewed last. For one thing, interviewers' techniques and procedures improve with practice, and the team should be working at its best before interviewing top managers. Second, any suspicions or doubts the team may have about information obtained from the first executives interviewed can be resolved by formulating specific questions for top management.

Preparing Questions for Interviews

Before the interviews are conducted, the study team prepares a list of general questions that apply to all interviewees. The team also may prepare specific questions for certain executives, as needed. The following general questions, while furnished *only as a guide*, usually apply to all interviewees:

1. Briefly, what is your area of responsibility?
2. What are its basic objectives?
3. What are the three greatest problems you have met in achieving these objectives within the last year?
4. What has prevented your solving them?
5. What is needed to solve them?
6. What value (in man-hours saved, dollars saved, or programs enhanced) would better information have in these areas?
7. In what other areas of your responsibility could the greatest improvements be realized, given the needed information support?

8. What would be the value of these improvements in man-hours saved, dollars saved, or programs enhanced?
9. How would you rate your information support with respect to adequacy, validity, timeliness, consistency, cost, and volume?
10. What is the most useful information you receive?
11. How are you measured?
12. How do you measure your subordinates?
13. What other kinds of measurement are you expected to make?
14. What kinds of decisions are you expected to make?
15. What major changes are anticipated in your area in the next year? Three years?
16. What do you expect to result from this study? What does it mean to you and to the business?
17. Do you have any additional thoughts or comments?

The interviewer must understand the purpose of the general questions in order to be sure that the needed information is elicited. If an interviewee's response does not provide the information desired, it may mean that the question should be worded differently. Let's examine the purposes of the general questions and kinds of responses needed:

- Questions 1-2. The interviewer asks the executives what their responsibilities are because conclusions drawn from the organization chart may not be correct. Also, the objectives that executives set for themselves in performing their functions may have a large bearing on their information needs.
- Questions 3-6. When asking about problems, the team should be sure that the executives do not indicate only their last problems. Sometimes people overlook more serious problems that occurred months before.
- Questions 7-9. The information satisfaction and needs questions should focus on the additional costs that may be incurred from inaccurate and untimely information. The value of any additional information desired should also be determined.
- Question 10. The team must determine where the current information support is adequate so that these benefits are retained.
- Questions 11-13. The questions concerning measurement are intended to disclose how the effectiveness of resource allocation and use is measured. A major factor in business success is efficient use of resources.
- Question 14. The question on what decisions are made is intended to disclose whether executives receive adequate information on which to base them.
- Question 15. The questions on anticipated changes are important because major information systems development cycles can take from two to five years to complete. Recommendations for information systems development could be impacted by plans to centralize, decentralize, change the organization, introduce new products, etc.
- Question 16. The questions regarding expectations for the BSP study may help the study team ascertain how likely management is to accept and implement the study recommendations.
- Question 17. Additional thoughts and comments make an appropriate way to terminate the interview.

Determining Team Member Interview Roles and Interviewee Characteristics

One of the tasks that must be performed when preparing for interviewing is to assign roles for team members to play during the interviews. There are several factors to consider when making these assignments.

The interview team is drawn solely from study team members and may include IBM representatives. Regardless of the number of people on the study team, there are usually no more than four active team members present at any particular interview.

A given team member need not play the same role in each interview. Furthermore, some members are usually assigned to more than one role in an interview. The following roles should be filled:

- An individual to present the study objectives and approach to the interviewee. This person may have another role in the interview as well.
- An interviewer.
- A backup interviewer to listen carefully to the proceedings to make sure that all questions are asked and that all answers are understood. The backup interviewer asks any additional questions necessary.
- Someone to take notes during the interview and have them organized, typed, and presented to the interviewee afterwards for approval.
- A supplementary note-taker if the situation requires.

Sometimes the interviewee is a member of a department in which a team member serves. Because of the possible sensitivity of some of the information, it may not be advisable for that team member to be an interviewer during that session. Perhaps the individual should not even be present. Similarly, if there are any personality clashes between an interviewee and a particular team member, that team member should probably not be presented at the session.

In determining who will serve in what roles, consideration should be given to individual abilities. Some people are better speakers than others; some are fast

and accurate note-takers. The interviewer for any session should be chosen for knowledge of the area of the business to be investigated and for ability to achieve rapport with the interviewee.

Before each interview session, the study team should meet to discuss various matters that are specific to that interview. The responsibility of individual team members during the interview has already been covered. The study team should also review:

- Background of the interviewee
- Responsibilities of the interviewee
- Process in which the interviewee is involved
- Data processing support provided to the interviewee's area of responsibility
- Specific questions to ask the interviewee, which may be based on prior interviews with other executives

Conducting Interviews

The best location for interviews is the study control room, where interruptions are least likely and where the matrices and charts can be displayed on the walls for reference. If the control room is close to executive offices, it should be convenient for the interviewees.

The following sequence of activities will normally transpire after the interviewee arrives and has been introduced to the interview team:

- The interviewee should be walked through the wall charts by the interviewer and he should be asked to validate the data on the charts. There should be agreement between the interviewee and the study team on the business process definitions and their relation to the organizational entities, particularly within the interviewee's area of the business. Make changes to the wall charts if necessary. (See Appendix G.)
- The interviewee is seated so that he can see the wall charts during the interview. This should make the interviewee more comfortable and thus enhance support.
- All the general questions and any specific questions previously formulated are asked. Many questions are likely to arise in the course of the interview and these are also asked.

As indicated before, the interviewer often needs to ask more questions than were prepared. It is impossible to predict all the possible responses that an executive might offer. From statements made during the interview, new questions are bound to arise. Furthermore, questions often must be formulated or restated during the interview to elicit the exact information needed. This is particularly true when trying to secure value statements that can be quantified, especially when discussing intangibles.

Following are some techniques that can be used to prompt an interviewee to express values quantifiably:

- Have the interviewee express a potential benefit as a percentage.
- Have the interviewee express a potential benefit in terms of a range – for example, a saving of $200,000 to $400,000.
- If the interviewee gives an exceptionally high value that might not be acceptable to other managers, suggest using 50 percent of that figure.
- In a particular operational area, cite a well-known industry average.
- Determine the company's present position in the industry and ask what it would be worth to be in a stronger position.
- Employ in-depth logical questioning. If the value of a particular benefit cannot be quantified by the interviewee, try to break down the various costs of not having the benefit.
- Determine the objectives against which the executive is measured and what it would be worth to achieve those objectives.
- Determine what the company is currently spending, what the forecasts are, and how these spending levels would be affected if a certain capability were available.

The interviewer must be sensitive to the responses of the executive in order to probe more deeply into any subject area when appropriate. Team members should be attentive to the executive's tone of voice and facial expressions, which are often a key to underlying feelings.

Preparing Interview Summaries

The study team should keep current with the documentation of the interviews. Immediately after the interview has been concluded, the two study team members who were taking notes during the interview should pull the notes together and have them summarized and typed.

Summaries should be formatted so that problem statements and key points appear in separate paragraphs. Each problem statement and key point can then be easily extracted for later analytical work. The summaries are reviewed by the team leader, then presented to the interviewee for approval. The interviewees should approve and return them to the study team within two days after the interview so that all team members can keep informed. Copies should be given to all the team members and one copy filed as the official study team document.

In addition, any changes resulting from the interview should be posted to the wall charts to keep them as accurate as possible.

As the interview summaries are returned, the study team members can begin to assess the business problems and/or opportunities (see Chapter 10).

Chapter 10. Assessing Business Problems and Benefits

The reduction and use of data obtained during the BSP interview process is extremely important. This chapter presents a structured and proven procedure to organize interview data and provide a useful format for later use in determining information architecture and priorities. Figure 25 presents a logical flowchart of the data reduction procedure and is keyed to the text by step number.

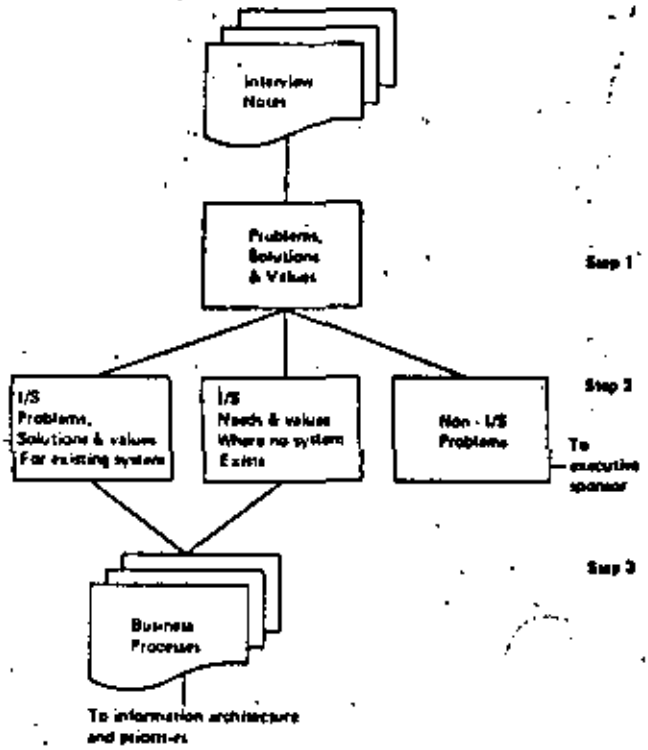


Figure 25. Data reduction steps

Step 1: Summarize Interview Data

The team selects out of each set of interview notes (1) the problems, (2) solutions (if given), (3) value statements – tangible or intangible – if given, (4) the processes impacted, and (5) the processes causing the problems.

Problems as stated by the businessman may be "perceived problems" (that is, perceived from the impact they make on a particular process) or they may be actual root-cause problems. "Perceived problems" are necessarily the starting point for problem assessment in BSP. A perceived problem is something a businessman sees as a problem: It may turn out to be a real problem, an incomplete version of a real problem, or just a symptom of one.

In order to assess problems objectively, therefore, perceived problems must first be traced back to root causes and forward to end-effects. The loss represented by each end-effect may then be quantified as far as possible, and the relative contributions of root causes to that loss may be assessed.

Figure 26 illustrates the worksheet to be used in this step. The team, as a unit, then considers the individual analyses and drafts a composite list of problems, solutions and values. The team must take care to select only stated problems and not implied problems. If the team decides that an implied problem is of a magnitude and importance that it should be further clarified, communication with the interviewee on a formal or informal basis should be undertaken in order to establish the specifics and/or magnitude of the "implied problem." If the communication establishes additional significant data not in the original notes, it must be added as an addendum to the validated original notes with reference to the follow-on communications. A copy of the addendum must be forwarded to the interviewee for his validation.

The entire composite list should now be typed with double spacing between entries. Each entry must be code-connected to the original interviewee. Assigning a number or letter or combination of same to each interviewee allows his identity code to be attached to each entry, while maintaining confidentiality.

Step 2: Classify Interview Data

Using the interview notes and worksheets, the team now examines each line entry and establishes whether it is in the category of:

1. I/S problems/solutions and values (existing systems)
2. I/S needs and values (no system exists)
3. Non-I/S problems

Sometimes a problem will have both an organizational solution and an I/S solution. Example: "There is a need for better marketing research information." This may require that a research group be established to gather information and an information system be developed to store and manipulate the information. At the end of step 2, the line entries extracted from the interview process should be in the three categories named above.

It is critical at this point that the "non-I/S problems" be logically grouped, stripped of all identity, and very carefully handled. It is not the mission of the BSP study team to solve the non-I/S problems. However, the information obtained in the interview processes, which relates to these problems, is important and should be properly presented to the appropriate individual (usually the sponsor) for his consideration and appropriate action.

Interviewee: Mr. H.R. Zimmer, Vice President of Production

Team Member: Mel Aksarben Date: August 24, 1978

Major Problem	Problem Solution	Value Statement	Information System Needs	Process/Group Impacted	Process/Group Causing
Lack of effective production planning impairs profitability	Mechanized production planning	Improve profit; improve customer relations; improve service of supply	Production planning	Production	Production
Increasing accountability of board of directors; demand for financial and management auditing	Better information	Increase value from outside directors	Cost system	Finance	Finance
Lack of "what if" capability on cost sheets	Mechanized online cost sheet capability	Improve profit; better customer relations	Cost system	Finance	Administration
Inability to look at enough alternatives in business planning	Financial modeling ("what if" capability)	Raise profit; curtail losses; improve growth	Planning simulation	Finance/management	Management
Lack of ability to identify and promote qualified people	Better information on personnel resources	Retain good people; improve morale	Skills inventory system	Human Resources	Human Resources
Lack of expertise in marketing impairs growth and profits	More marketing awareness and more people with marketing expertise	Satisfy customers		Marketing	Human Resources
Loss of ability to effectively transfer people; low productivity. Long training	Description of work to be done in each area	Manage people properly; improve quality of work; increase productivity	Order status	Human Resources	Sales Operations
Poor customer relations; loss of profit; excessive inventory	Better sales analysis and productive planning; better buyer reports	Improve customer relations	Order status	Marketing	Sales Operations
Poor customer relations; loss of profit; excessive inventory	Better sales analysis and productive planning; better buyer reports	Improve customer relations	Production planning	Marketing	Production
Poor customer relations; loss of profit; excessive inventory	Better sales analysis and productive planning; better buyer reports	Improve customer relations	Sales analysis	Marketing	Marketing
Smaller, more frequent orders; lead times too critical; higher cost of handling orders	Order trend analysis/costs	Control cost better	Territory analysis	Sales Operations	Sales Operations

Figure 26. Interview analysis and data reduction worksheet

Step 3: Relate Data to Processes

All of those interview entries which were grouped as I/S problems/solutions or I/S needs and values should now be cut from the worksheet. Example:

K-2 Lack of effective production planning impairs profitability	Mechanized production planning	Improve profit; improve customer relations; improve service of supply	Production planning	Production	Production

Next a flipchart page should be created for each business process group and its processes. These should be arranged around the control room at a level handy to "post" to. Team members then take each of the prepared interview entries and stick them onto the flipchart for the business process that causes the problem. If possible, the entry should be placed within the process most appropriate.

Don't worry about being too precise on the first "posting." Additional teamwork is required to finalize the procedures.

When all entries have been "posted," the team concentrates on one business process at a time to ensure that all entries have been posted to the right process. It is normal to have to go back to the original validated interview notes or even the original handwritten notes to firmly establish the topic area and background that triggered the interview comment (entry). The need for mobility of the item entries quickly becomes obvious during this fine-tuning step.

Summary

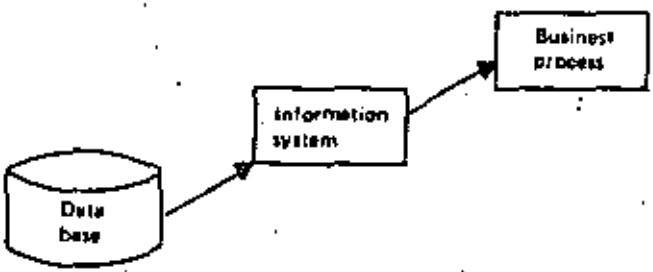
It is possible to group the interview data entries in several ways. They may be related to the causing processes or the impacted processes using matrices as illustrated in Figure 27 or in other summary formats as appropriate. This data will provide valuable perspective as the team develops recommendations on information architecture and systems priorities.

PROCESS GROUP / PROBLEM	Marketing	Sales operations	Engineering	Production	Materials management	Facilities	Administration	Finance	Human resources	Management
Market/Customer Selectivity	2	2								2
Quality of Forecasting	3									4
Order Entry Log		3					1			
Product Line Profitability	1							1		
Market Share Deterioration	3									2
Product Development Support			4					1		1
External Reporting Requirements							1	1		
Control of Orders		4	3	6			2			
Facility Planning			1	1		2				
Inventory Level Consistency				5	5					
Purchasing Control				6	2					
Control of Production Costs				2			5			
Labor and Machine Utilization				5						
Plant Performance Measurements				5						
Shop Manpower Utilization				2						
Asset Performance								1		
Variance Analysis							2	1		
Working Capital Management								2		
Manpower Development									1	
Job Descriptions									3	
Total	9	9	8	32	7	2	11	7	4	9

Figure 27. Problem/process matrix, showing the number of times the problems were stated

Chapter 11. Defining Information Architecture

Having acquired an understanding of the business processes and the data required to support them, the team should continue its analysis by determining how the data will be managed to support the processes. The data classes that have been identified can be logically grouped into data bases. Information systems then become the vehicles used to insert data into and extract it from data bases, and formulate useful management information to support the business processes.



In order to identify the information systems and subsystems areas to be developed, an *Information architecture* is defined, using a diagram that shows the relationship of data to systems and the processes supported by each. The architecture diagram, then, is the blueprint that outlines for each system area (1) the data created, controlled and used, (2) the relationship of system to system, and (3) the systems that support a given process. The architecture diagram allows us to take into consideration the data requirements of later subsystems at the time of developing earlier subsystems in order to maximize sharing of data.

First, the architecture is developed by evolving through several iterations of a process/data class matrix. Data flow is then added to the architecture, followed by subsystem identification and analysis of subsystems for prerequisites.

Developing the Architecture Diagram

Begin with the process/data class matrix where the process axis is in life cycle sequence based upon the key resource, as in Figure 28. Next, arrange the data axis by taking all data classes created by the first process and grouping them to the left, or origin of the matrix. Group all data classes created by the second process next to those created by the first process and continue this grouping until all data classes are covered. Figure 29 shows the result, with all "creates" grouped diagonally from the matrix origin to the opposite corner.

Although the next step is somewhat a matter of judgment, the object is to group the processes and data into major systems areas. This can be accomplished by looking at the process groups and the data created by them, then boxing in these groupings as in Figure 30. Where a process group creates no data, the box is arbitrary. The boxes represent logical subsystem groupings with responsibility for creating and maintaining specific related classes of data.

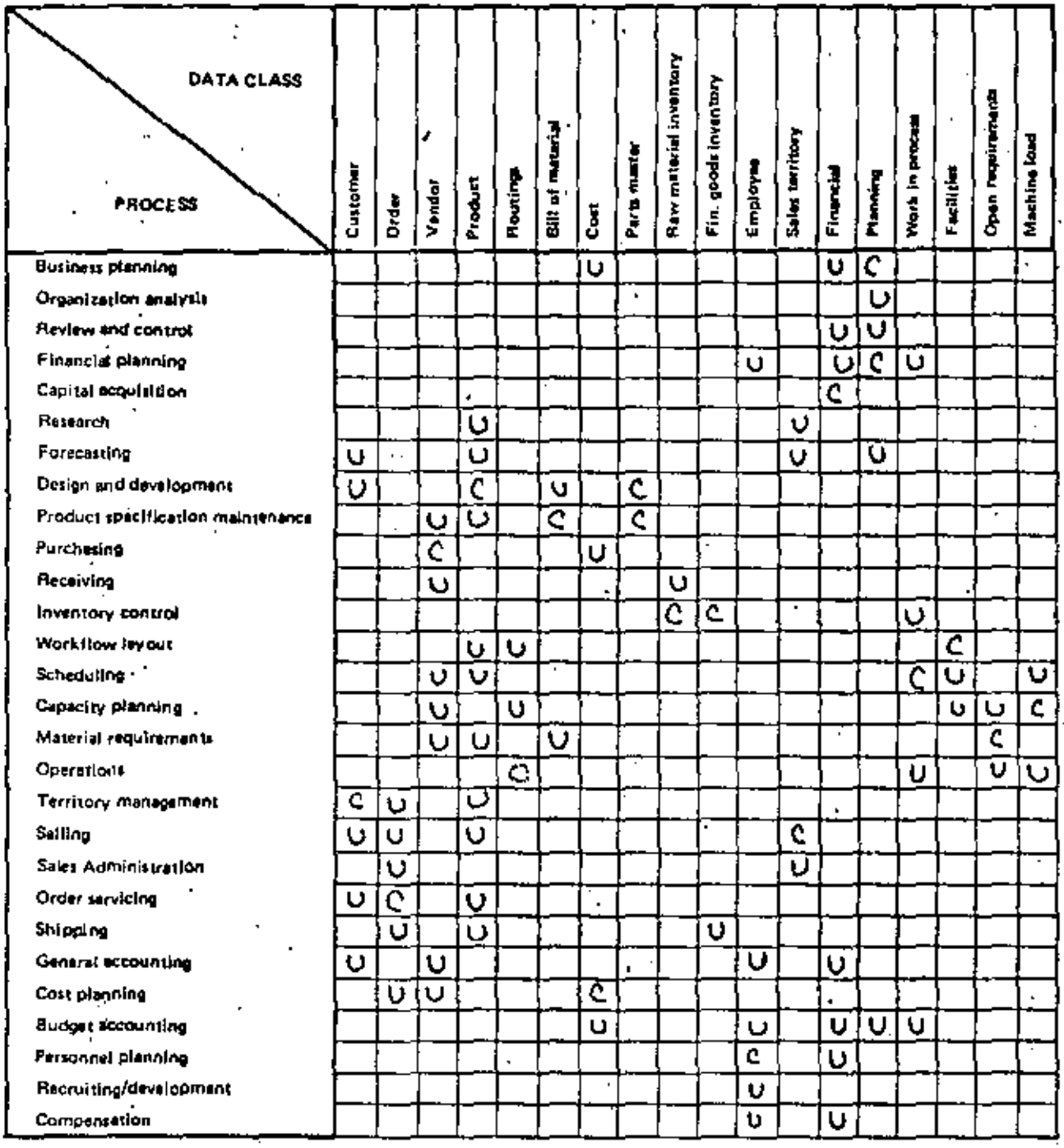


Figure 28. Data class by process, showing data creation and usage

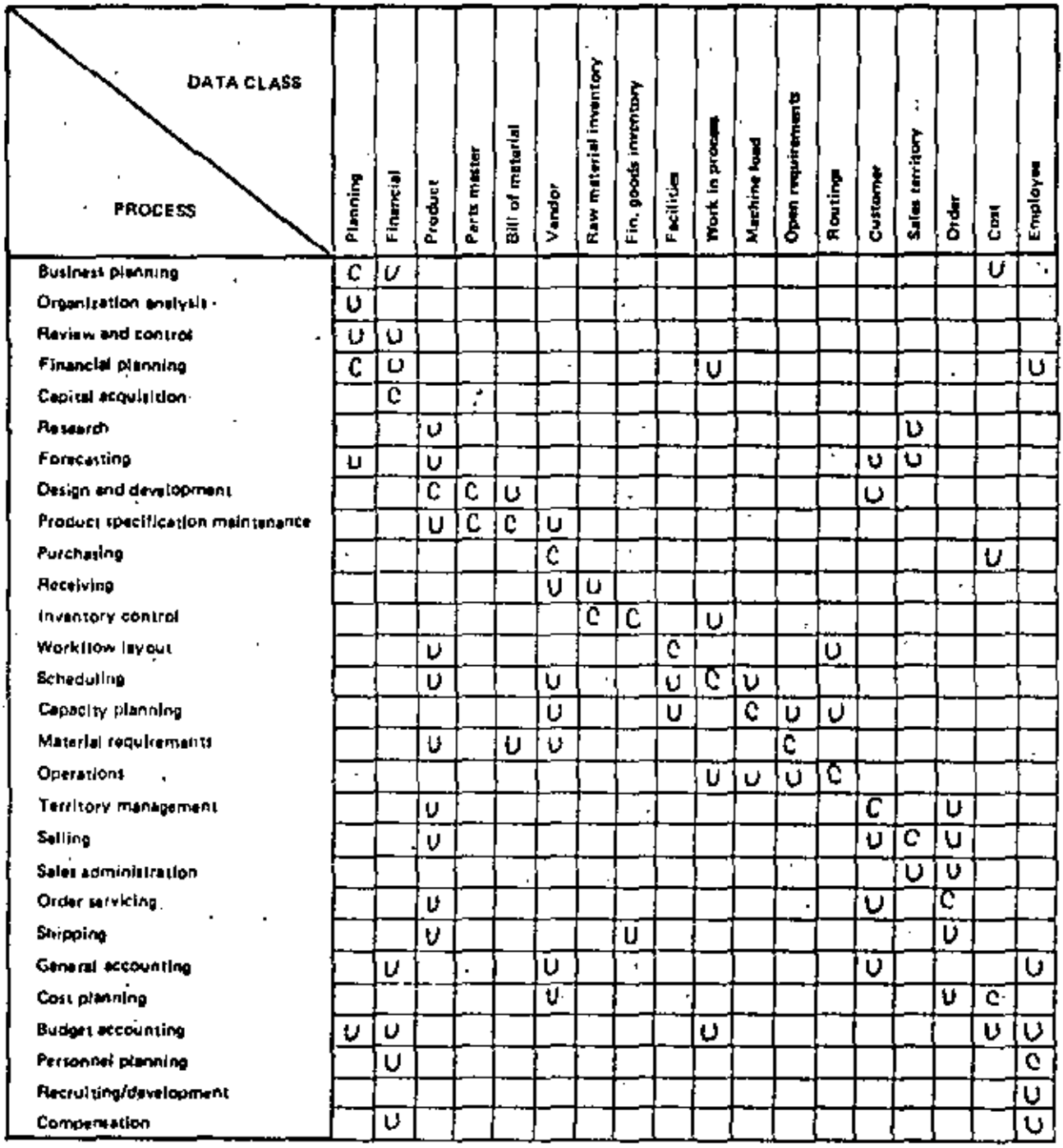


Figure 29. Data classes arranged by creating process

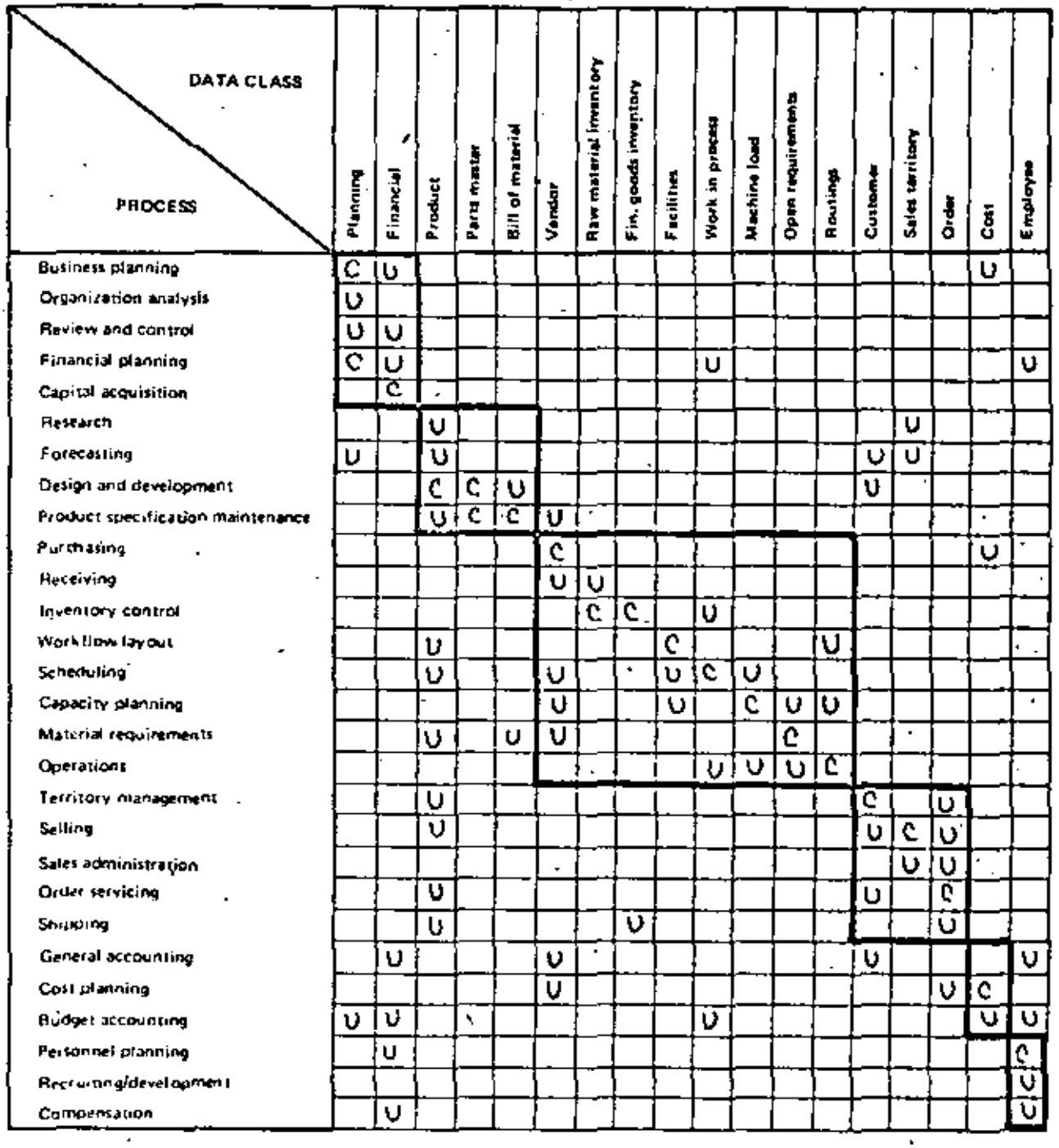


Figure 30. Process/data class groupings

Data Flow

Where the U falls outside a system box, a means to represent data flow from one system area to another is desirable. This flow is represented by arrows in Figure 31. The third system down uses "Product" data created by the second system. The arrow shows data flow from the second to the third system. The second system uses "Customer" data created by the fourth system, so an arrow indicates this data flow. Figure 32 shows all data flow from system to system represented by one-way arrows. This figure will be used later for subsystem identification.

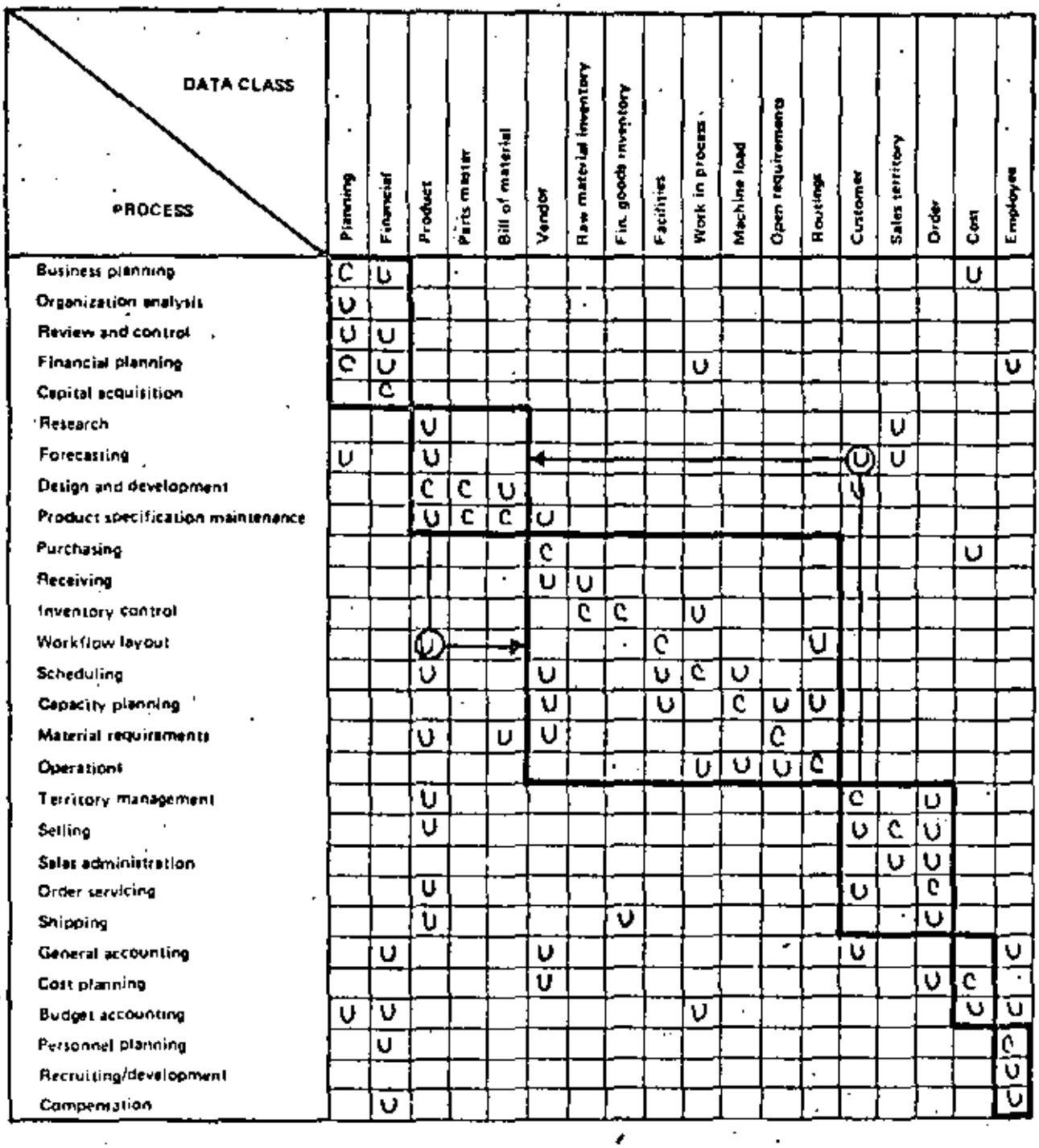


Figure 31. Data flow determination

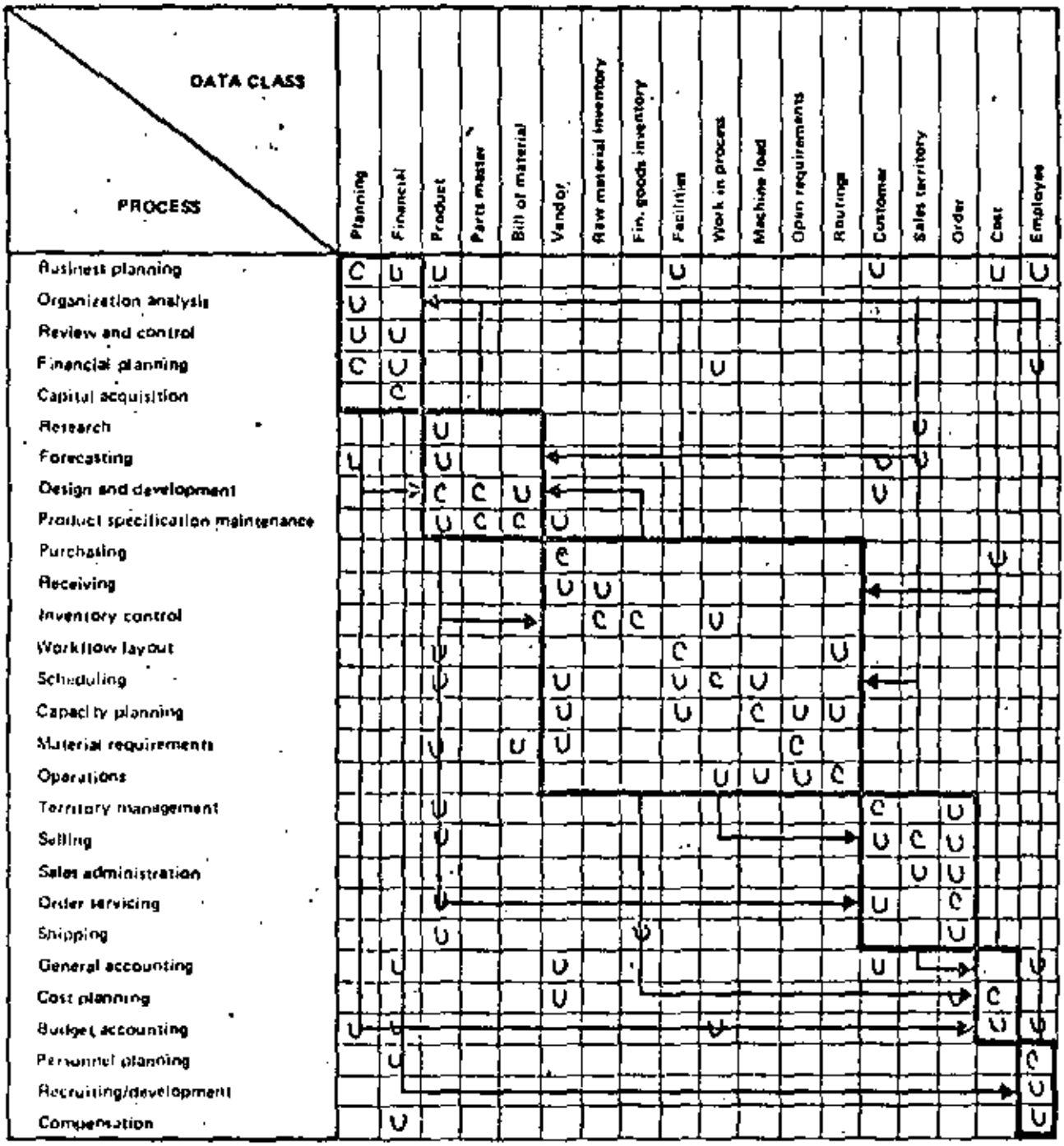


Figure 32. Data flow

With data creation and use indicated by the boxes and arrows, the Cs and Us may be eliminated and names applied to the major systems areas (Figure 33). While Figure 33 represents the completed information architecture, a final rearrangement of axes and use of two-way arrows allows preparation of a simplified graphic of the architecture. Figure 34 shows such a graphic with the same systems. Such illustrations have proved useful in management communications regarding the information architecture.

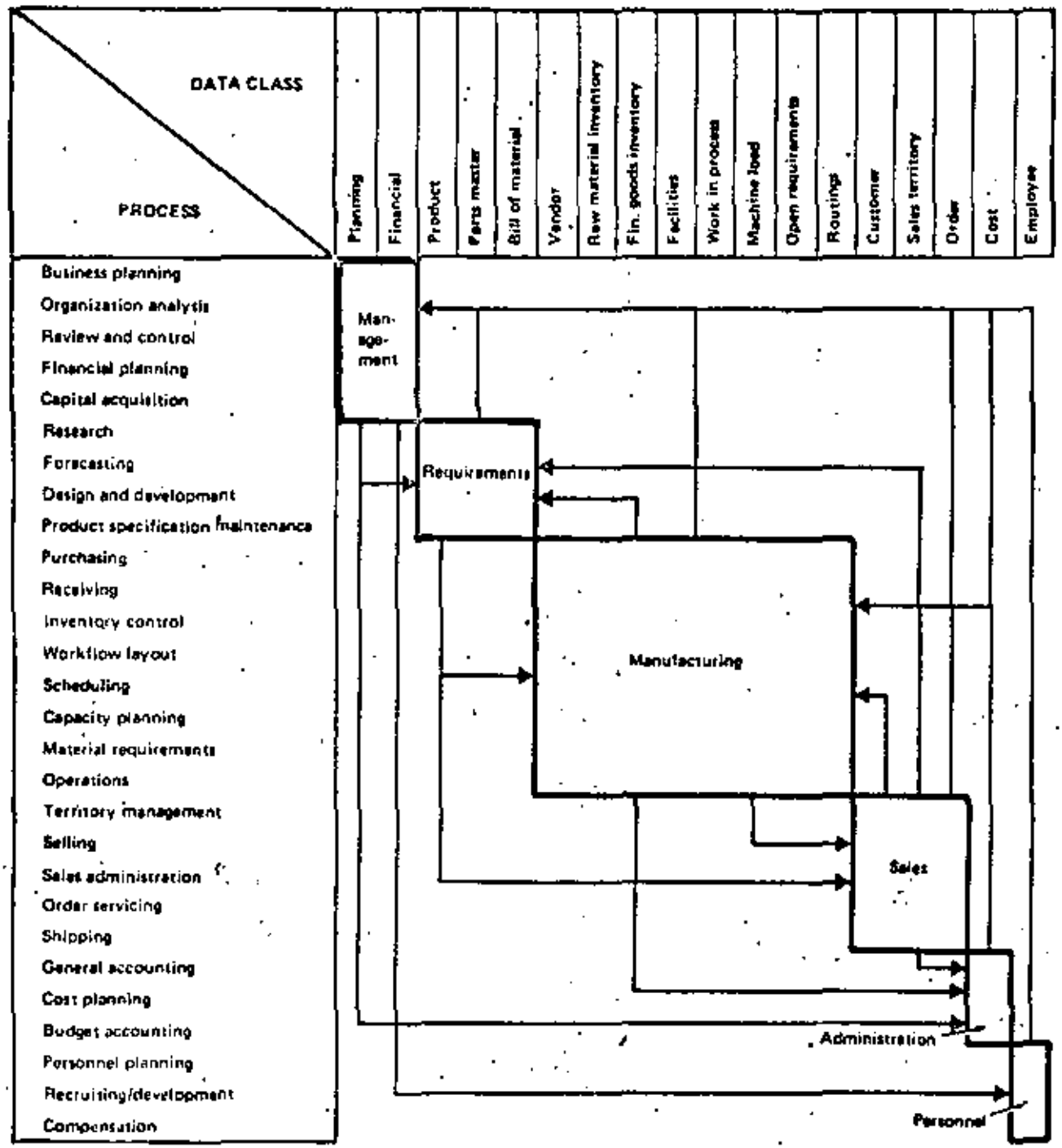


Figure 33. Information architecture

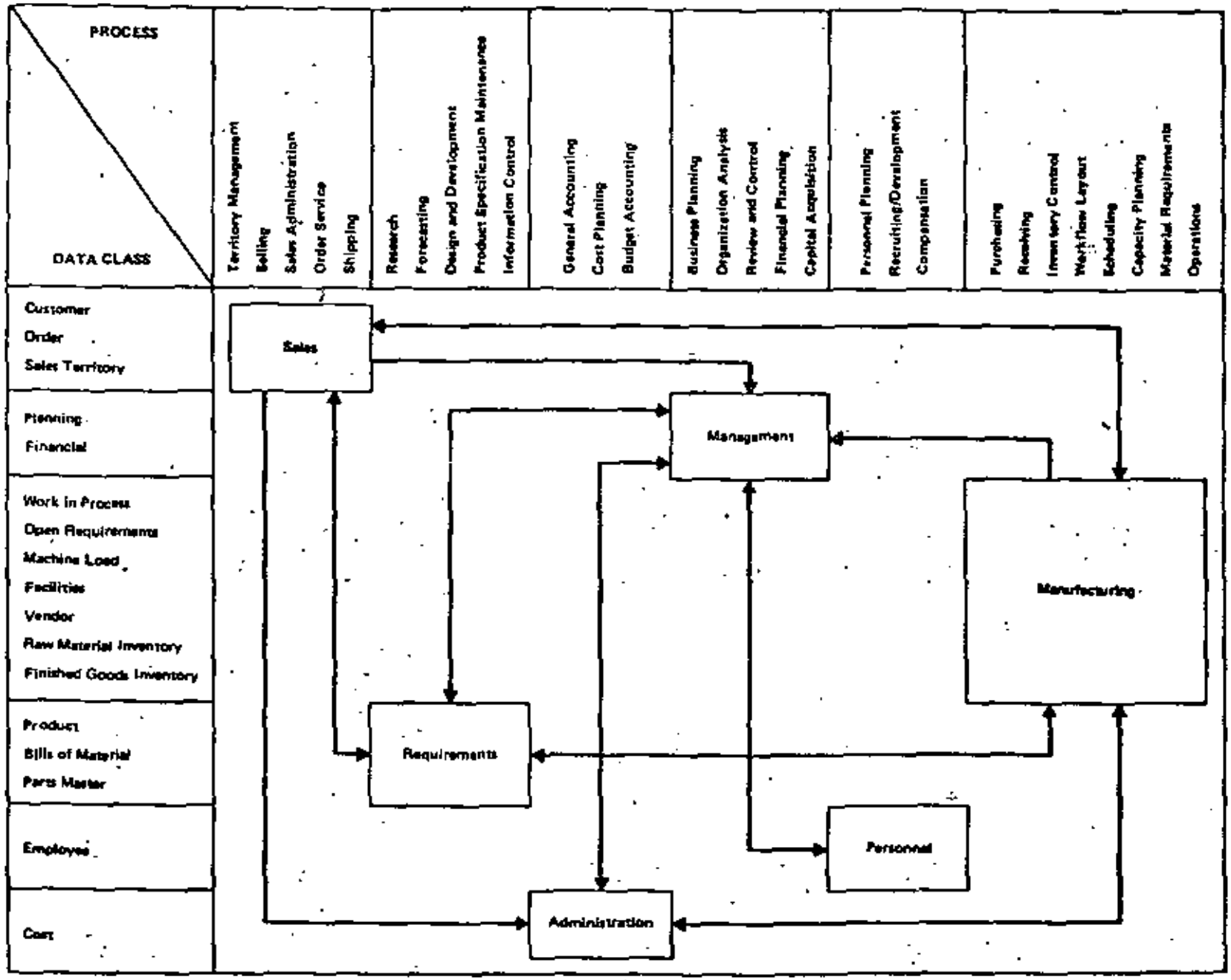


Figure 34. Graphic rearrangement of information architecture

Subsystem Identification

The first cut at identifying subsystems is again an arbitrary classification. Using the architecture development diagram such as Figure 30, define each C as a *create* subsystem. Define each U associated with a given process as a single usage subsystem.

With this starting point, logical combinations and/or splits can be made. Usage subsystems can be combined where use of the data is similar. Where create and usage subsystems are highly interdependent, a combination may be logical. Where a usage subsystem has dissimilar functions included, it can be split into two or more subsystems.

When the subsystems have been identified, a description of the functions of each (approximately one paragraph) should be written.

Analyzing for Prerequisites

With an outline of subsystems at hand, the final step is a prerequisite analysis, i.e., which subsystems must be in place before others can be created. By using the information architecture just developed and the team's understanding of the business, the interdependencies among subsystems can be analyzed. Once more, a matrix can be used to tabulate the results of the prerequisite analysis. The vertical axis of the matrix should list all subsystems, while the horizontal should list the prerequisite subsystems (Figure 35).

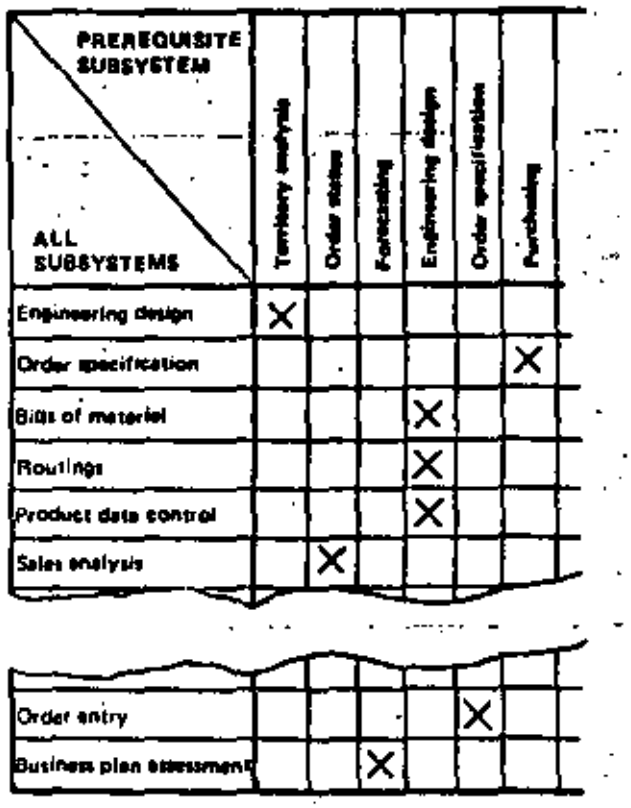


Figure 35. Prerequisite subsystem analysis

Architecture Use

The information architecture identifies the systems and subsystems with respect to the data they create, control and use, and with respect to the business processes they support. The interdependencies have been analyzed as a basis for the prioritizing and planning that now take place. The information architecture provides an excellent foundation for analysis of distributed information systems requirements and data base planning, subjects covered further in Chapter 16. As each step is taken in development of the information systems, the architecture provides a blueprint showing where each piece of the system fits and what its relationship to the other pieces is.

Chapter 12. Determining Architecture Priorities

The team should select and recommend to management a portion of the information architecture to be implemented first. The purpose of identifying and recommending such a follow-on project is to begin implementation as early as practical. This will establish a "pay as you go" foundation for implementation of information systems.

The activity of selecting first subsystem(s) should rely primarily on a value analysis activity following the interviews. Its purpose is to produce a priority sequence of subsystem(s) that were identified in the information architecture activity of the BSP. The tasks to be performed are:

1. Determine selection criteria and document the technique to be used
2. Apply criteria to the subsystems
3. List subsystems in priority sequence
4. List dependencies or projects that are prerequisite to the first subsystems
5. Document or describe the recommended subsystems

Determine Selection Criteria

Since data is now being treated as a business resource, t/s projects should be able to be evaluated by management in the same way other business projects are evaluated. Therefore, it is recommended that the team use whatever justification technique presently exists within the organization for evaluating new projects. This will ease decision making as executive management makes the necessary trade-offs between, say, developing a new product, expanding a facility, making an acquisition, or implementing a major information system.

Some of the questions to be answered in determining the first subsystem(s) are:

- Will the subsystem provide a significant near-term saving and a substantial long-term return on investment?
- Whom will it impact, and how many people will be involved?
- Will it lay the groundwork for an initial data base structure for the architecture?

A method of determining logical priorities is to group the major criteria into four categories:

1. Potential benefits (value judgments)
   - Tangibles
   - Intangibles
   - ROI
2. Impact
   - Number of organizations and people affected
   - Qualitative effect
   - Effect on accomplishing overall objectives
3. Success
   - Degree of business acceptance
   - Probability of implementation
   - Prerequisites
   - Length of implementation
   - Risk
   - Resources available.
4. Demand
   - Value of existing systems
   - Relationship with other systems
   - Political overtones
   - Need

Apply Criteria and List Systems

The major systems comprising the architecture can then be analyzed as potential first-subsystem candidates and ranked on a scale of 1 to 10 for each of the four categories above. Some type of pictorial representation can then be drawn to emphasize the most needed subsystems. Figure 36 is an example of this analysis.

List Prerequisites

During the architecture development activity, the team prepared a matrix to identify system prerequisites (Figure 35). Once a first system has been identified, the team should review that matrix. Any steps or projects necessary for successful implementation of the selected first system should be sequenced and listed. This will be necessary for developing a meaningful action plan. Prerequisite projects should then be clearly described and their relationship to the selected first system defined.

Document Recommended System

Finally, the recommended first system (or first subsystems) should be further documented. The system should be described in sufficient detail that the executive can properly evaluate it. The description should include an overview of system functions, major purposes and processes supported. It should identify any new technologies and/or special skills that need to be acquired or developed in order to implement or operate the recommended new systems. A general description of perceived benefits anticipated should also be included. The recommendation, documentation, and presentation should be in a format consistent with standard company practice.

Because of resource limitations, the definition of more than the very highest priority systems may be impossible initially. However, the same method can be used in making later selections. After the implementation of the first system has been completed, the priorities of the remaining system should be reassessed. For example, after the first four systems have been implemented, the system initially listed fifth may no longer be at the top of the list because the requirements and problems of the business have changed.

Expansions and Variations

Another method that may be used for priority selection is to focus primarily on financial justification. Using this technique, the team attempts to focus on tangible benefits as provided in the value statements during the executive interviews. This technique, called risk-benefit analysis, is described in Appendix H.

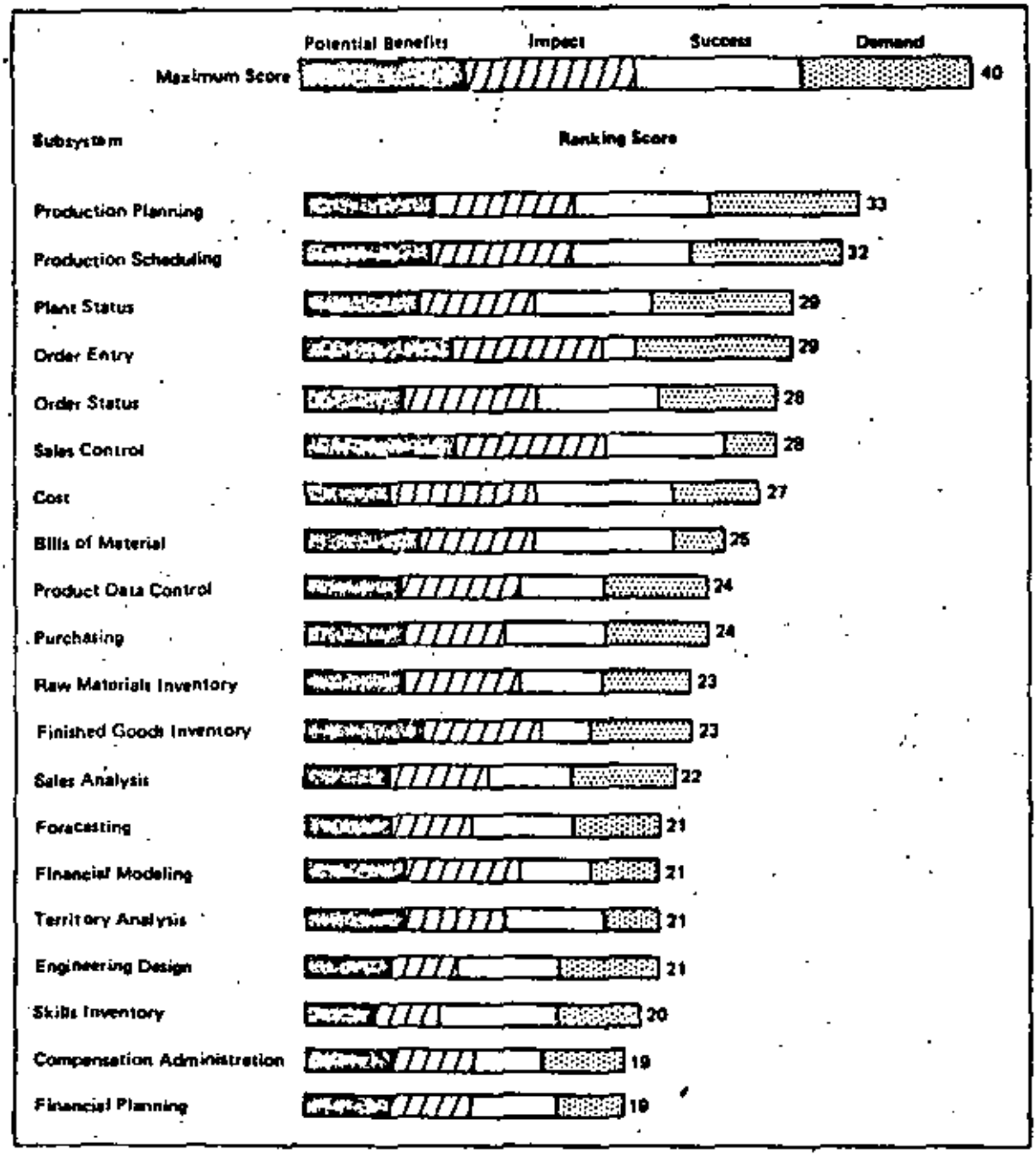


Figure 36. Sample subsystem ranking

Chapter 13. Reviewing Information Systems Management

The development and implementation of the information architecture cannot be accomplished without adequate planning, measurement, and control in the information systems functions. This chapter deals with those activities required to adequately review the Information Systems Management (ISM) function so as to recommend immediate changes and to specify ISM activities that are part of the follow-on projects.

The BSP study will not encompass all the ISM functions. The ISM analysis should be at the level of the I/S director's view and should include only the planning and control aspects of the major I/S processes. To put the ISM activities into perspective, this chapter starts with a definition of ISM and then goes on to the objectives of the ISM module in the BSP study.

Purpose of ISM in the BSP Study

The purpose of Information Systems Management is to manage the I/S resources for the most efficient and effective support of the business. The management system helps provide the continuous planning, control, measurement, and operation of the information systems function. An example of the total spectrum of ISM processes is represented by Appendix I.

ISM as Part of the BSP Study

The objectives of the ISM module are to:

- Study the facts relating to ISM produced in other modules of the BSP study
- Understand the findings and conclusions on I/S support of the business in relation to ISM
- Further specify I/S objectives
- Ensure that the first subsystems selected will be supported by the current or planned ISM functions
- Identify those ISM projects that have high priority and should be follow-on ISM projects

ISM Inputs

There are actions in other study modules prerequisite to performing the ISM activities in the BSP study:

- From study preparation:
  - I/S mission and objectives
  - I/S plans and strategies
  - I/S measurements
  - I/S organization
  - I/S budgets
  - I/S trends (resources, expenses)
  - I/S inventory and locations (equipment, software, systems, personnel)
- From determining business requirements:
  - Description of business objectives
  - I/S support problems (from executive interviews)
  - Major ISM problems/requirements from DP director's I/S review)
- From determining architectural priorities:
  - New technologies and skills required to support the development of the first subsystem(s)

Flow of Activities in the ISM Module

Figure 37 gives an overview of the activities involved in the ISM module. The activities in italics are not part of this exercise but are included to help the reader see where ISM fits with the other activities. As noted in the figure, the required ISM changes are developed through three channels:

1. I/S objectives
2. Information architecture
3. Problem analysis
   - I/S problems
   - ISM problems

The changes required by each of these are then consolidated into a list for developing ISM priorities, which serves as input to the ISM portion of the action plan resulting from the BSP study.

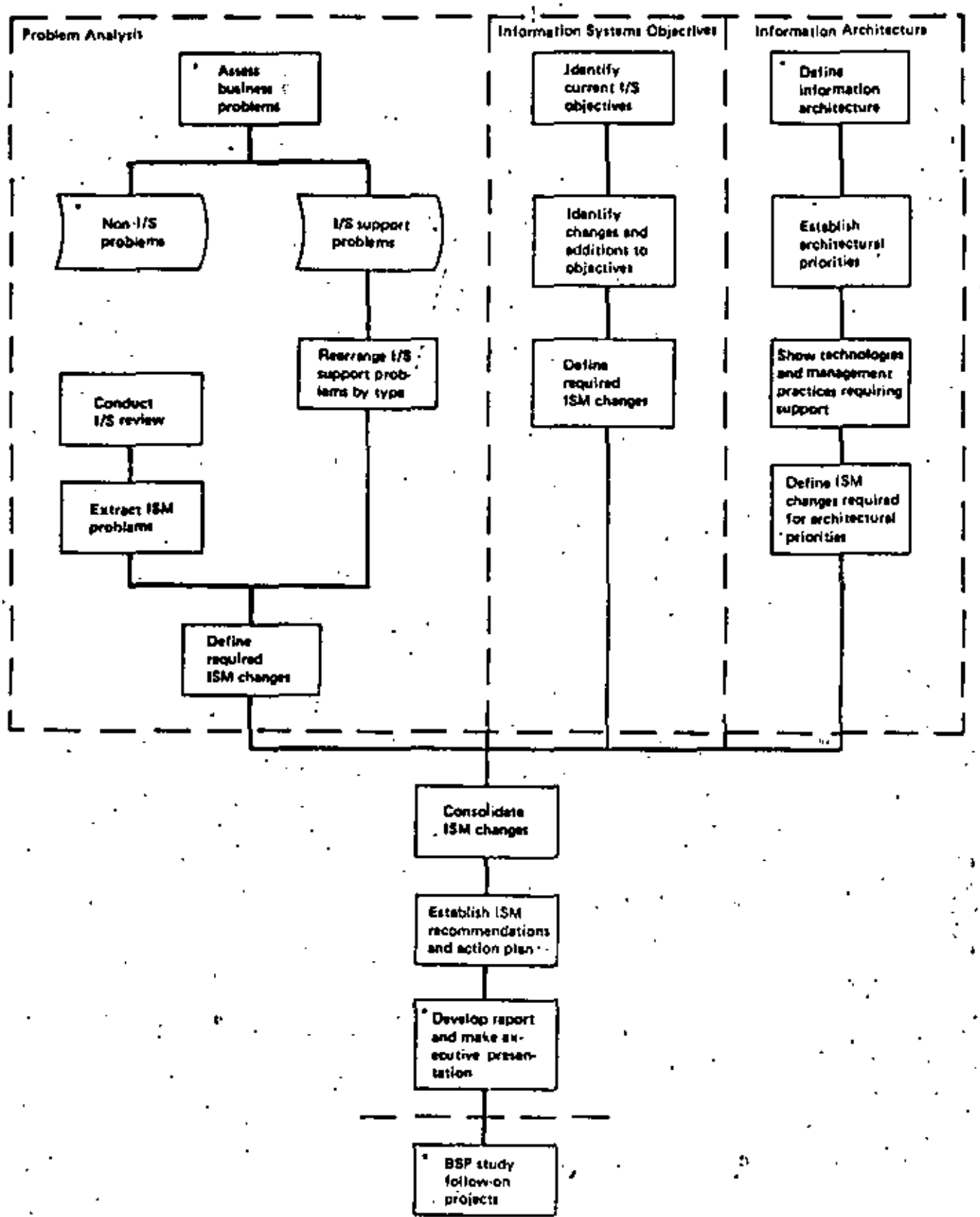
Compiling the I/S Objectives

The I/S objectives are required to provide the framework to provide consistency in the development and employment of I/S resources in support of the business. I/S objectives must be consistent with the objectives and management systems of the business so as to serve as a guide for the development of I/S plans and strategies in the BSP follow-on activities.

The current I/S objectives were documented in the BSP study preparation phase and discussed in the I/S director's presentation.

In determining the present I/S objectives to be modified and additional objectives that should be defined, the following list can be used as a guide. The objectives can be specified by defining the long-term view of each of these categories in information systems.

1. I/S Personnel
   - Organization
   - Skill mix
   - Development and training



*Not involved in this exercise but added to the chart for purposes of perspective on ISM

Figure 37. Flow of ISM in a BSP study

2. I/S Finance
   Appropriations
   Expense budget
   User budget for I/S
3. User
   Capability
   Responsibility
      Development
      Data management
4. I/S Facilities
   Processing sites (data center - user locations)
   Equipment
   Software
   Communications network
5. Applications
6. Data

The following is a sample of I/S objectives following this approach:

Personnel

1. Provide staff assistance for planning and control to all levels of I/S line management.
2. Centralize major development and control functions with decentralized operations.
3. Develop I/S personnel who are specialists for each function of the business.

User

1. Provide maximum data processing capability and data access to users while maintaining central control.
2. Establish user responsibility for information requirements and validity of I/S outputs during the external design phase.

Following this procedure in compiling the I/S objectives should result in implied changes to ISM. These changes should be listed and reviewed with the I/S director and then held for later consolidation in the establishing of ISM project priorities.

Ensuring ISM Supports Architectural Priorities

As Figure 37 shows, the architectural priorities are a major input in determining the changes to ISM. The first data bases and information systems will probably require new technologies that must be provided and managed. They may also require new management practices. For example, a move to distributed data processing may require new practices regarding funding, chargeouts, control, and education. List these changes and save them for the consolidation.

Problem Analysis

Rearrange I/S Support Problems by Type

The third channel for identifying ISM changes is problem analysis. The problems that have been identified in the BSP study are separated into I/S support problems and non-I/S problems. Although the I/S support problems may have been aligned with business processes and with data, they should be rearranged at this time for better evaluation of their solutions through changes to ISM. The rearrangement should result in the categorizing of the problems by the following ISM processes for ease of defining and implementing changes: strategic planning, management control, I/S development, systems management, data administration, and organization and administration. Categorizing the problems by ISM processes will also provide direct input to the follow-on activities.

After rearranging the I/S support problems by type, check to see that those resulting from inadequate or nonexistent systems have been addressed by the proposed information architecture. This may identify changes to existing systems. Such information would be given to the I/S director for appropriate action. The other problems should be translated into required ISM changes. For example, the inaccessibility of existing data may require a change to the user function, giving the user direct access to selected data through his terminal. This would be in line with the objective to distribute more data processing capability to the users and require additional controls. After listing the required ISM changes, hold them for combining with the ones resulting from the next step.

Extract ISM Problems from I/S Review

During the kickoff, the director of information systems, or his representative, gave a review of the status of information systems in the company. Part of that review covered current problems in information systems management. There is a significant difference between these problems and those in the preceding paragraph. The I/S support problems may imply problems in ISM or may be attributable to other causes, whereas specific ISM problems may be spelled out during the I/S review. Categorize this set of problems also by information systems processes.

Because one of the major BSP objectives is the development of a long-range information systems plan, it should receive major emphasis during problem analysis and it must, of course, be one of the follow-on projects. Of equal importance with I/S planning is the subject of data management. Since data has been identified as a corporate resource, a major emphasis must be put on its management. This also will be one of the follow-on projects in ISM; both it and I/S planning will be elaborated upon more fully in Chapter 16.

Consolidating and Prioritizing ISM Changes

Consolidate all of the implied ISM changes from the three sources. Develop criteria to be used to rank the ISM changes, such as impact, potential benefits, resour-

ces involved, and probability of successful implementation. Apply weights for each factor and determine the total weight and priority associated with each of the ISM changes. This selection process should parallel that used in establishing the architectural priorities.

Developing Recommendations and Action Plan

Although the above activity will give priorities, do not establish schedules until this list is combined with actions required for other follow-on activities. The developing of recommendations and an action plan is covered in the next chapter.

ISM Outputs of a BSP Study

The following list summarizes the outputs that should result from the ISM portion of the BSP study:

1. Findings and conclusions
   - Major ISM problem analysis
   - ISM problems deduced from the I/S support problems
2. I/S objectives
3. Definition of high-priority ISM projects
   - First systems support requirements
   - Projects addressing major ISM problems
   - Immediate actions recommended
4. Action plan for follow-on projects

Chapter 14. Developing Recommendations and Action Plan

Specific BSP study recommendations may center on the areas of:

1. Information architecture
   a. Detailing the subsystems of the information network
   b. Interim improvements to current systems
2. Information systems management, including:
   a. Data management to control the data resource of the organization
   b. Information systems planning process to ensure that the information architecture remains responsive to changing conditions
   c. Measurement and control to provide a system of procedures, controls and structure for future implementations
3. Architecture priorities – development of the first system to be implemented

For each recommendation, there may be an associated project, and for every project an action plan should be developed identifying the key decisions and activities required to help management provide proper direction. The action plan should give the project manager a firm basis for a smooth transition from the study work product to the particular project. The plan should describe the costs, potential benefits, and schedules for each project in enough detail that an informed executive decision can be made to approve or reject the project.

Each action plan should include the following:

Scope of project. Describe subject of project, including size and purpose.

Deliverables. List and define each output or result expected of the project.

Schedule. Identify expected start and concluding dates as specifically as possible.

Potential benefits. Describe anticipated benefits or reasons for doing the project.

People and skills. Describe manning of project in terms of management as well as staffing.

Tools and techniques. Describe any required practices or methods as specifically as possible.

Training. Describe specific orientation, education, and study necessary for successful execution of the project.

Communications. Identify coordination, liaison, interface points, or functions as well as documentation requirements.

Logistics. Define materiel and facility support necessary and indicate when it is needed.

Control. Define project control methods and responsibilities as well as review/approval requirements.

The amount of emphasis placed on the recommended follow-on projects varies greatly from one organization to another. Normally, multiple follow-on projects will run concurrently. The number of simultaneous system projects varies with the needs of the organization, the resources and time available to perform each project, and other factors.

The completion of the activities above enables the BSP study team to proceed to its final activities: preparation of the study report and presentation of the study results to executive management. Thorough preparation of the action plans, potential benefits, and costs enables the executive sponsor to evaluate the recommendations. The study team can then ask for and expect a prompt approval of the recommendations and a prompt commitment by management to implement them.

Chapter 15. Reporting Results

Having defined the information architecture, identified the architecture priorities, reviewed and assessed I/S management, and developed recommendations and the action plan, the BSP study team is ready to complete its mission by preparing the study report and preparing and delivering an executive presentation. The purpose of the report and presentation is to obtain further executive management commitment and involvement for implementing recommendations from the BSP study.

The following activities are performed in preparing the study report and executive presentation:

- Report outline is prepared
- Report is written
- Executive presentation medium is determined
- Executive presentation is prepared and delivered

Report Outline

During the initial steps of preparing for the study and developing a work plan for conducting the study, the team prepared a preliminary report outline. Since the report now materializing from that outline is to be a consensus of the entire team, each of its sections should be reviewed by each of the members so that the final document reflects all their comments.

The report may be structured in many ways, depending upon precedent and methods of presentation within the business conducting the study. However, to assist the team in its consideration of pertinent areas, an extensive list of topics is presented in Appendix J.

The most significant findings, conclusions, and recommendations should be summarized in the first few pages of the report for the use of top management. Supporting details should be included later and in the appendices for other members of the organization and for team members who will participate in future follow-on activities.

Report Preparation

The primary writing responsibility for each of the sections of the report was assigned during study preparation. As the study progresses, changes, additions, and deletions can be made to the preliminary outline. By the time the executive interviews have been completed, agreement should have been reached on the final table of contents, and the individual study team members should have available to them most of the information needed to complete their assigned sections. The study team leader usually assumes the responsibility for writing the background and overview because much of this material comes directly from the orientation information presented to the team at the kickoff meeting.

The conclusions, recommendations, and an action plan should be reviewed with the executive sponsor before the team drafts the final report. Controversial areas or areas of high impact on the business may be reviewed with the executive involved, to determine how best to present the recommendations.

Presentation Medium

The principal factors to consider in determining how the report should be presented are the type and size of the audience and the accepted ways of making such a presentation in the particular business environment. Consultation with the executive sponsor early in the study to obtain his advice on this subject can be most helpful in establishing the proper direction.

If the presentation is made to a small group, easel flipcharts are adequate and popular. If the audience numbers more than a dozen, viewgraphs or slides should be considered. If slides are to be used, adequate time should be allotted to have them prepared, whether in-house or by an outside vendor.

Executive Presentation

The executive presentation can be developed completely from material contained in the final draft of the report. The principal aims of this presentation are to inform management of the study's findings, make recommendations, and secure approval of the action plan.

The presentation should be concise – preferably no longer than an hour. It should be logical and factual and should end with recommendations for the follow-on activities. For example, it may take the following form:

1. Introduction
   - Background and overview
   - Objectives
   - Scope
   - Study team
2. Study approach
   - Business and I/S review
   - Business processes
   - Business/information systems relationships
3. Major problem identified
4. Conclusions and recommendations
   - Information architecture and priorities
   - I/S management requirements
5. Action plan for follow-on project activities
   - Description(s)
   - Deliverables
   - Resource requirements
   - Schedules

Chapter 16. Overview of Follow-On Activities

The BSP study ended with the development of an action plan for the follow-on activities and a presentation for management approval to proceed in the execution of that plan. This final chapter of the BSP Planning Guide gives an overview of the follow-on activities and serves three purposes:

1. To relate the follow-on activities to the BSP study
2. To show how to capitalize on the results of the BSP study
3. To further explain the major follow-on projects

The following section helps in putting the follow-on activities in perspective and in emphasizing their importance. It also explains how to get the maximum use of the output from the BSP study. The latter part of this chapter attempts to give enough of an understanding of the follow-on projects to allow for scheduling them in the action plan and to help preparing for them.

Perspective on Follow-on Activities

Relation of BSP Study to Follow-on Activities

The follow-on activities are a continuation and expansion of the major activities in the BSP study. The major thrust of the study was one of understanding, developing findings and conclusions, and making recommendations. While the thrust in follow-on projects is still to understand, greater emphasis is put on detailed definitions and planning.

There is a great need for communications among the project teams if project implementation is to be successful. Therefore, the information systems director should take overall responsibility for the projects, since the functions performed during the follow-on activities are a part of the ISM responsibilities. Because the follow-on activities build upon the results of and the information gathered in the BSP study, there is a need for continuity of team members. At least one person on the BSP study team should have been chosen from the data processing function, with the idea that he would remain for the follow-on activities.

The functions performed by the executive sponsor for the BSP study should be performed by a steering committee for the follow-on projects, unless there is a management committee that will perform them.

Business processes continue to be the vehicle for understanding detailed business requirements and for further defining the information architecture with special emphasis on the definition of the first system.

Preparation for Follow-on Activities

Because of the variations in the findings and conclusions that may result from BSP studies, the follow-on activities and the emphasis placed on each of them will vary from one BSP study to another. For purposes of this discussion, the following assumptions are made:

1. ISM functions must be changed or added to provide a controlled environment for the development and implementation of the information architecture identified in the BSP study. A detailed information systems plan will be developed that will reflect these planned changes.
2. The information architecture must be further defined and data bases identified.
3. A first system has been chosen for development and implementation.

Orientation of Teams

Before the beginning of each of the follow-on projects, the I/S director should provide orientation for project team members. Since many of the team members will not be familiar with BSP, its principles and objectives should be summarized. It is very important that everyone on the project teams understand exactly how the BSP study was conducted and become familiar with the outputs so as to be able to build upon them.

A review of the BSP study report for the areas to be studied in the follow-on projects will help to determine whether team members need special education. A specification of tools and techniques may also reveal the need for special education. If required, appropriate classes can be scheduled on an individual basis. If the I/S director was not a member of the BSP study team, it might be advisable for him to attend the IBM Information Systems Planning course.

The following is a sample list of topics for project team orientation:

- Introduction by the sponsor
- Objectives of the orientation
- Information systems concepts and perspectives
- Data base concepts
- Overview of the BSP study methodology
- Detailed discussion of the BSP study
  - Business environment
  - Objectives
  - Major processes identified
  - Major problems
  - Current I/S support for the business
  - Interviews
  - Information architecture
  - Major findings and conclusions
  - Recommendations
- Introduction to the follow-on activities
  - Relation to the BSP study
  - Relation to the BSP study deliverables

Discussion of deliverables from the follow-on activities
Explanation of follow-on projects
Introduction of tools and techniques
Introduction of preparation activities
Activities already performed
Activities to be performed by team members
Interviewing strategies and selection of interviewees
Use of matrices
Pilots of forms, tools, techniques and interviews
Introduction to HIPO and its use in the follow-on projects
Assignment of responsibilities
Development of task descriptions by team members
Explanation of project control systems

Information Systems Management

ISM is the keystone of effective support of the business by information systems. It can be defined as the management of the information systems resources for the most efficient and effective support of the business. The importance of continual emphasis on ISM is summarized by the following statements:

- Development of the information architecture will take place over a period of years.
- Changes to business strategies and plans as well as I/S technology will be continual during the development of the information architecture.
- The processes of managing I/S will have to be refined and changed continually to properly plan, measure, and control required I/S resources.
- The BSP study is a one-time effort that should be followed by continual I/S planning to fully capitalize on the results.

ISM in Perspective

As covered in Chapter 13, the ISM follow-on projects will emanate from (1) requirements to support I/S objectives, (2) changes resulting from the development and implementation of the first system, and (3) recommended changes to solve I/S support problems and ISM problems.

From this it is safe to assume that the recommendations included:

1. Emphasis on information systems planning and control and on data administration.
2. Some close-in projects, such as the establishment of a steering committee and refinements to project control. Capacity planning is also a natural follow-on to a BSP study since it examines the data processing capacity and determines whether changes are required to accommodate the architectural priorities or to solve some of the I/S support problems.
3. A major project(s) in ISM, e.g., establishing an action plan to move to distributed data processing.

Perhaps the most important action to be taken after the BSP study is to decide what person or group of persons is responsible for continuing the I/S planning using the BSP study as a base. While the BSP study culminated in an approved action plan, the overall objective of an ISM project will be to develop a long-range I/S plan that will direct the design, development, and implementation of an information architecture. It should include sufficient detail on projects, resources, and schedules to guide all levels of management on what is to be done, when, and by whom in the organization. Since the long-range I/S plan is an integral part of the business plan and is prepared with full management involvement, it should help to provide that the I/S resources of the business will be used to effectively support the business.

Also, since data management is a complex area requiring long lead times, the data administrator should be decided upon as early as possible so that this function can be properly coordinated.

The importance of I/S planning and data administration cannot be overemphasized, and they will nearly always be a part of the follow-on activity. The activities will not be explained further here since they are covered thoroughly in IBM, GUIDE, SHARE, and trade publications.

Some of the close-in projects should be done immediately after the BSP study recommendations are approved and before most of the follow-on activities are started. The establishment of a steering committee and a project control system are excellent examples since a steering committee should be available to direct the follow-on activities and a project control system should be in place to help the I/S director adequately control and coordinate the activities.

The major ISM follow-on project will probably be a long-range activity such as I/S planning or data administration or a combination of both.

Because of the size of these projects and their interactions with all the I/S processes, it is important to fully understand all the functions of ISM before undertaking any recommended change to any part of ISM, just as it was important to fully understand the business before starting to fulfill the information requirements. Some mechanism should exist to get the broad view of ISM, and BSP can be used as that mechanism. You may treat I/S as a business-within-a-business and apply to it the same BSP techniques that you have applied to the business in which it is located.

Information Systems as a Business

Many of the activities for performing a BSP study in the I/S function have already been accomplished:

- Facts relating to information systems were gathered as part of the study preparation phase.

- ISM responsibilities, activities, and problems were identified by the I/S director during the kickoff.
- Many of the findings and conclusions have already been put together as a result of examining the I/S facts and the problems.
- A first ISM project has been identified and an action plan established.

While some of these activities will have to be expanded, they make a significant start in understanding the whole I/S function. The following identifies other steps that should be performed:

1. Defining ISM processes. Appendix I suggests the processes that take place in ISM. This may be used as a checklist to identify the ISM processes. Otherwise, you may wish to identify the information systems resources and the activities and decisions found in their management cycle. The following is a list of resources to be considered:
   Programs and systems
   Functional modules
   Data bases
   Hardware/software/communications facilities
   User facilities (terminals, languages, data bases, etc.)
   Physical facilities
   Finance
   People
2. Analyzing relationships of organization, processes, information systems, and data. This is an exact parallel to the development of the matrices to show relationships in the business BSP study, except that the information systems are those that support the I/S functions, such as equipment utilization systems and chargeout systems. The data is that needed to support these and other systems used to manage the I/S function.
3. Interviewing. The interviewing in ISM will normally be with the upper two levels of information systems managers, selective interactive users, the executive to whom the I/S director reports, and selected functional users.

The remaining steps are the same as the BSP methodology for the business and should be conducted by a small team led by the individual responsible for I/S planning or another person with a broad responsibility in I/S.

The BSP study of the I/S function may be done as part of the ISM follow-on project, as suggested. It should also be considered as an effort that can be made either concurrently with the business BSP study, before it, or immediately after it and before the start of any follow-on projects.

As an activity before the business study, the I/S study provides experience in the methodology, offers an excellent view of I/S as input to the business study, allows an early start on ISM projects, and can serve as a selling tool if some executives are skeptical of the methodology. It must be recognized that a major deficiency exists in the understanding of part of the I/S environment, which in this case is the rest of the business.

BSP is so demanding of a team's time that running the two studies concurrently will require two teams, closely coordinated. The added emphasis on ISM may detract from the major goal of understanding the business and may give the entire study a much greater flavor of an I/S project than is desired. The major advantage is the ability to relate I/S problems directly to ISM problems and make sound recommendations on ISM in the action plan and report of the business BSP study.

Ideally, the I/S study should be done immediately after the business study, since (1) the required inputs are readily available, (2) the momentum will carry over from the business study, and (3) the best environment will exist for conducting all the follow-on activities.

Information Architecture

The amount of follow-on effort required in the architecture area depends on the level of detail in the BSP study. If the information architecture was not completely defined as outlined in Chapter 11, the basic architecture definition should be completed as the beginning of a follow-on project.

The architecture refinement to be accomplished in a follow-on project should include a confirmation of the major I/S groupings, the subsystem breakout of each system, subsystem interrelation, and data flow among subsystems.

Several other activities should be included in the information architecture project:

- Examination of current data processing systems to determine how they can evolve (provided they can at all) into the new systems architecture
- Establishment of a continuing link between the architecture and the business plan to ensure that the I/S plan will be based upon a viable information architecture
- Documentation of alternative architectures that were tried or investigated and rejected, with the reasons for rejection
- Evaluation of the technical implications of the information architecture, including control systems, DB/DC requirements, and potential distribution of both information and systems

Architecture Refinement

Using the information architecture defined in the BSP study, a description of each subsystem should be prepared. Each description should include the purpose of the subsystem, business problems addressed, data

created and used, dependencies on other subsystems, general requirements for implementation, and priority of the subsystem with respect to all other subsystems.

Current Systems Examination

For data processing systems currently in use or under development, a careful examination should be conducted to determine how each one relates to the subsystems described in the information architecture. Possible modifications to current systems should be documented with respect to interfacing them to the future architecture. Where modified systems could be used in lieu of new subsystems, a description should be included in the architecture documentation. For systems already under development, an evaluation should be made of possible modifications to make the developing system compatible with the new information architecture. Analysis should be made of the impact on current development schedules versus the avoidance of modifying the system after implementation, keeping in mind present commitments and future priorities.

Data Base

Integral to information architecture definition in the BSP study were the data classes supporting the business processes. A logical grouping of related data classes will yield the set of data bases that will support the architecture implementation.

A data base can be defined as a nonredundant collection of interrelated data items processable by one or more applications. Some of the terms in the definition itself may need explaining:

- *Nonredundant* means that individual data items appear only once (or at least less frequently than in normal file organizations) in the data base.
- *Interrelated* means that the files are constructed with an ordered and planned relationship that allows data elements to be tied together, even though they may not necessarily be in the same physical record.
- *Processable by one or more applications* means simply that data is shared and used by several different subsystems.

Development of a data base has some obvious benefits. By consolidating files, the user can obtain better control of data and reduce storage space and processing time. Equally important are the resultant data synchronization and timeliness. Use of a single information source makes processing more accurate because all subsystems refer to the same data.

A data base system can help overcome some of the complexities of data management. It can provide additional data relationships while minimizing storage redundancy. Figure 38 illustrates how a data base system can support subsystem needs independently and currently.

A comparison of the data base environment with the traditional approach to systems development and maintenance reveals the advantages of the data base concept. In the traditional approach, a system is usually designed, programmed, tested, and then implemented as a total entity. Its advantages cannot be realized by the end user until the entire system is completed. The amount of time involved can cause frustration, since business requirements cannot be kept frozen long enough to avoid changes, delays, and false starts. Also, when data or logic changes are required, considerable testing may be necessary to determine how the change affects other programs or systems functions.

By contrast, the data base approach allows a gradual transition from existing systems to online, transaction-driven systems. The data base is created gradually, with just a few transactions implemented at a time. Current data base techniques facilitate this type of data base and system implementation. With gradual implementation of transactions, user department signoff can be obtained more easily and the user can enjoy the benefits earlier. This approach also helps overcome the problem of not being able to freeze business requirements and technological advances during the development cycle. Changes that must be made along the way impact individual transactions or become new transactions themselves. Changing data needs can be accommodated without affecting programs that do not use that specific area or segment of the data base. Thus, the data base environment can be a better way to accommodate change, deliver benefits to the user, and control development costs.

Distributed Information Systems

The general purpose of distributed information systems is to provide the user with a level of computing resource best suited to his operational requirements. The elements that can be considered for distribution include hardware, program execution, program development, and data. Degree of distribution can vary from totally centralized to totally decentralized. The information architecture defined in the BSP study can provide a good foundation for the additional planning necessary to address I/S distribution requirements.

If the information architecture follow-on project is to add distributed I/S parameters to the architecture, the project team must:

- Thoroughly understand the degree of distribution of each element
- Understand the arguments for and against distribu-

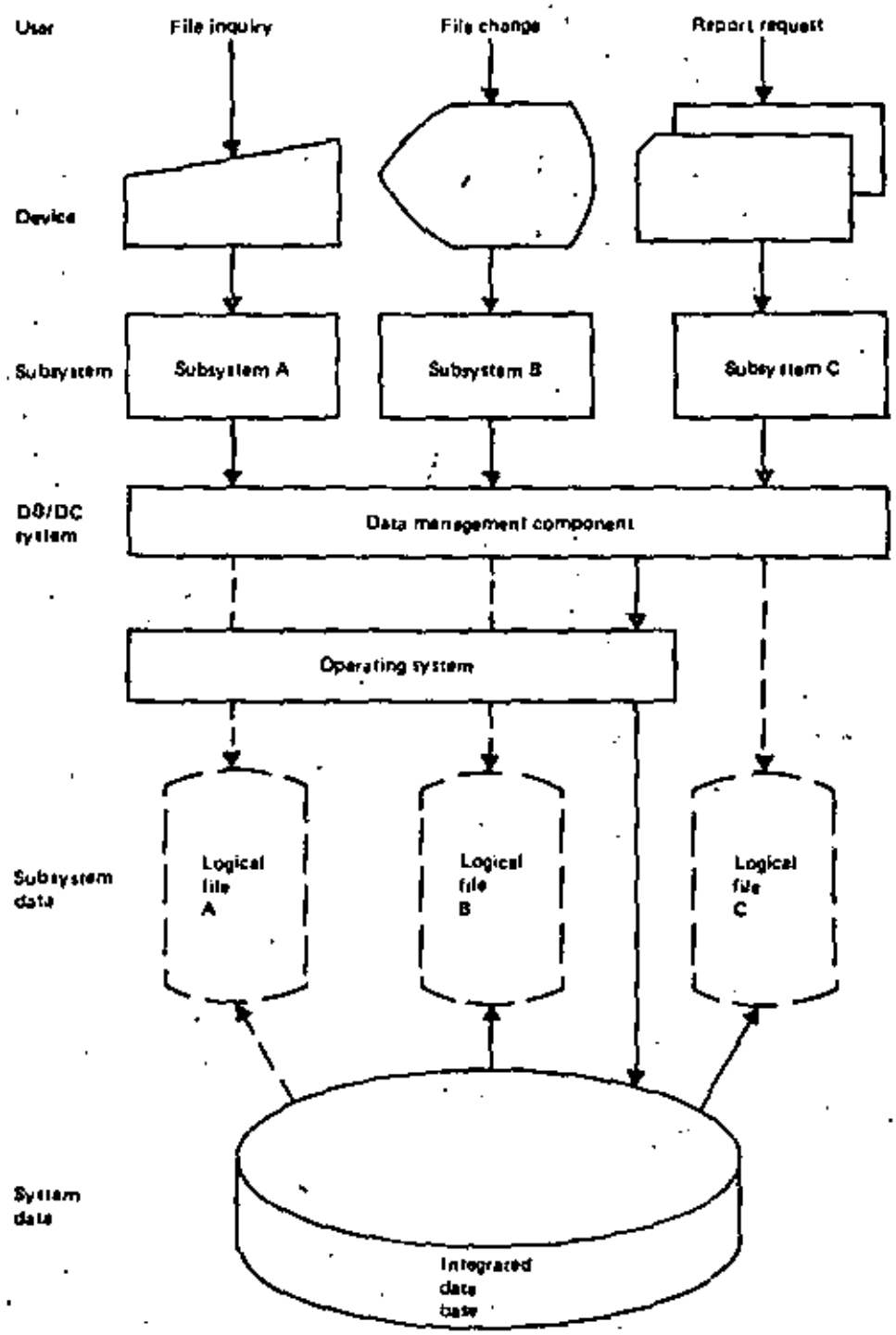


Figure 38. Data base system

tion of the various elements
- Be able to put into perspective all the factors that affect distribution and sharing of data processing resources
- Appreciate the need for a long-range I/S plan and for development of a facilities plan to support the I/S plan

Several of the matrices developed in the BSP study can be very useful in developing the architecture extensions with respect to distribution. The process and data flow analysis should be extended to include physical locations for systems and data requirements. Location/organization, location/process and location/data matrices should be prepared. A process/data matrix for each location (or group of similar locations) should be prepared with additional parameters on the axes to include data use, security, auditability, data occurrence (all or partial), activity volumes, response required, and criticality. Figure 39 shows this type of matrix for a manufacturing plant site.

These additions to the architectural description of the information systems can then be input to the individual subsystem requirement studies to determine detail distributed systems requirements. They will also give an overall feel for the applicability of distributed systems to the information architecture.

Developing the First System

The BSP team has recommended that a particular high-priority system be developed and implemented first. As development of that system begins, care should be taken that this system is not developed "by itself." The BSP study has set a certain direction for implementation of systems. This first development and implementation must support the business processes and must:

1. Fit properly into and interface with the other (future) systems of the overall architecture described in the BSP study
2. Provide for proper implementation of the appropriate data base(s)

As described in the ISM section of this chapter, these concurrent projects must be carefully managed (ISM) to ensure that they lead toward an integrated whole rather than perpetuate "separatist" or "standalone" applications development.

System development is a complex process. It requires technical expertise, effective communications, and skillful handling of conflicting objectives of users, management, and data processing.

Technical Factors

There is no doubt that developing information systems is more complex today than in the past. Factors such as data security, data integrity, data communications, rapid availability of large amounts of data, and backup and recovery requirements complicate the tasks that must be performed on a project. They also add to requirements for skilled personnel to perform such tasks.

In many areas technical risks may be reduced by incorporating appropriate hardware/software products, productivity techniques, and procedures. An investment of time in evaluating existing products and applying them to the recommended system will reap ample rewards in a better system implementation.

Effective Communication

Effective communication is indispensable to successful system development, which entails the communication and translation of management and user business requirements into achievable data processing solutions.

Only users can determine the business requirements of the system to be developed. Their involvement and continued participation are, therefore, essential.

Developers sometimes try to second-guess the needs of users and to create systems they believe will do the job. More often than not, this approach is unsatisfactory. While the *technical* aspects of such a system may be sound, the real *business* needs of the organization may not be addressed or met.

Users and developers need a mutual understanding of what is to be done and a commitment to their respective responsibilities. The development process itself must ensure communication and commitment between the two groups.

Communications may be greatly facilitated by the use of standardized documentation techniques such as HIPO, structured walk-throughs, and the like. These not only reduce misunderstandings but contribute to increased productivity, which in turn should speed up implementation of the system or application. Where appropriate, such aids should be evaluated, adopted, and incorporated in the development project.

Priorities and Conflicts

The need to set priorities for conflicting objectives is crucial. Conflicts can make even simple situations complex and can cause many serious project difficulties.

Even before a project is under way, the new system must compete with the demands of trained personnel to maintain old systems. Application development resources are in short supply, and experienced development skills may be hard to find because everybody has a backlog of applications waiting to be developed.

The project manager, therefore, must be qualified both in project management techniques and in the business area to be supported by the first system.

Location: Plant Site / DATA CLASS / PROCESS	Product	Bill of Material	Parts Master	Routings	Planning	Facilities	Employee	Vendor	Raw Material Inventory	Finished Goods Inventory	Work in Process	Machine Load	Open Requirements	Customer	Sales Territory	Order	Financial	Cost		Security	Auditability	Volume	Responsiveness
Design and Develop.		A	A											A								L	L
Product Spec. Maint.		A	A					A												C		L	M
Information Control	A			C																	C	L	M
Planning																							
Research																							
Forecasting																							
Financial Planning					A		A				A						A					L	L
Capital Acquisition																							
Funds Management	A						A									A	A				A	M	M
Workflow Layout	A			A		C															C	M	H
Maintenance						A					A											L	L
Equip. Performance						A														A		M	M
Personnel Planning							C										A			C	C	M	L
Recruiting/Development							A															L	M
Compensation							A										A			A		L	L
Purchasing								A										A				H	H
Receiving								A	A													M	H
Inventory Control									C	C	A										C	M	H
Shipping	A									A						A						H	M
Scheduling	A					A		A			C	A								A	C	M	H
Capacity Planning				A		A		A				C	A							A	A	M	H
Material Reqmts.	A	A											C								C	M	H
Operations				A							A	A	A									L	H
Territory Mgmt.																							
Selling																							
Sales Administration																							
Order Servicing	A													A		A						H	H
General Accounting							A	A						A			C				C	H	L
Cost Planning								A								A		A				L	L
Budget Accounting					A		A				A						A	A				M	L
Business Planning					A												A					L	L
Organization Analysis					A																	L	L
Review and Control					A												A					L	L
Risk Management																	A					L	L
Use	A	A	A	C	A	C	C	A	C	C	C	C	C	A		A	C	A					
Security	A				A		A	A						C	A	A	A	C					
Auditability	A				A																		
Data Occurrence	T	T	T	T	T	T	P	P	T	T	T	T	T	P		P	P	T					
Currency	I	I	S	S	L	L	L	O	I	I	I	S	O	O		S	L	O					

Use and Audit: A = Access only C = Access and Change
Security: A = Unauthorized Access protection
C = Unauthorized Change protection
Occurrence: T = Total record needed
P = Partial record needed
Currency: I = Immediate; S = Same day
O = Overnight; L = Longer
Volume/Response: H = High/quick; M = Medium; L = Low/slow

Figure 39. Distributed information requirements analysis

Ample education, documentation, and products are available for both. A thorough analysis of support should be made and a technical support plan should be put together to utilize the education, documentation, and products wherever possible. You should work with your IBM marketing representative to develop such a plan.

The project manager and appropriate team members should be thoroughly prepared in these areas before the start of the project. This should result in effective project management, which in turn will reduce risks and optimize the business value of the first implementation.

DIVISION DE EDUCACION CONTINUA
FACULTAD DE INGENIERIA U.N.A.M.

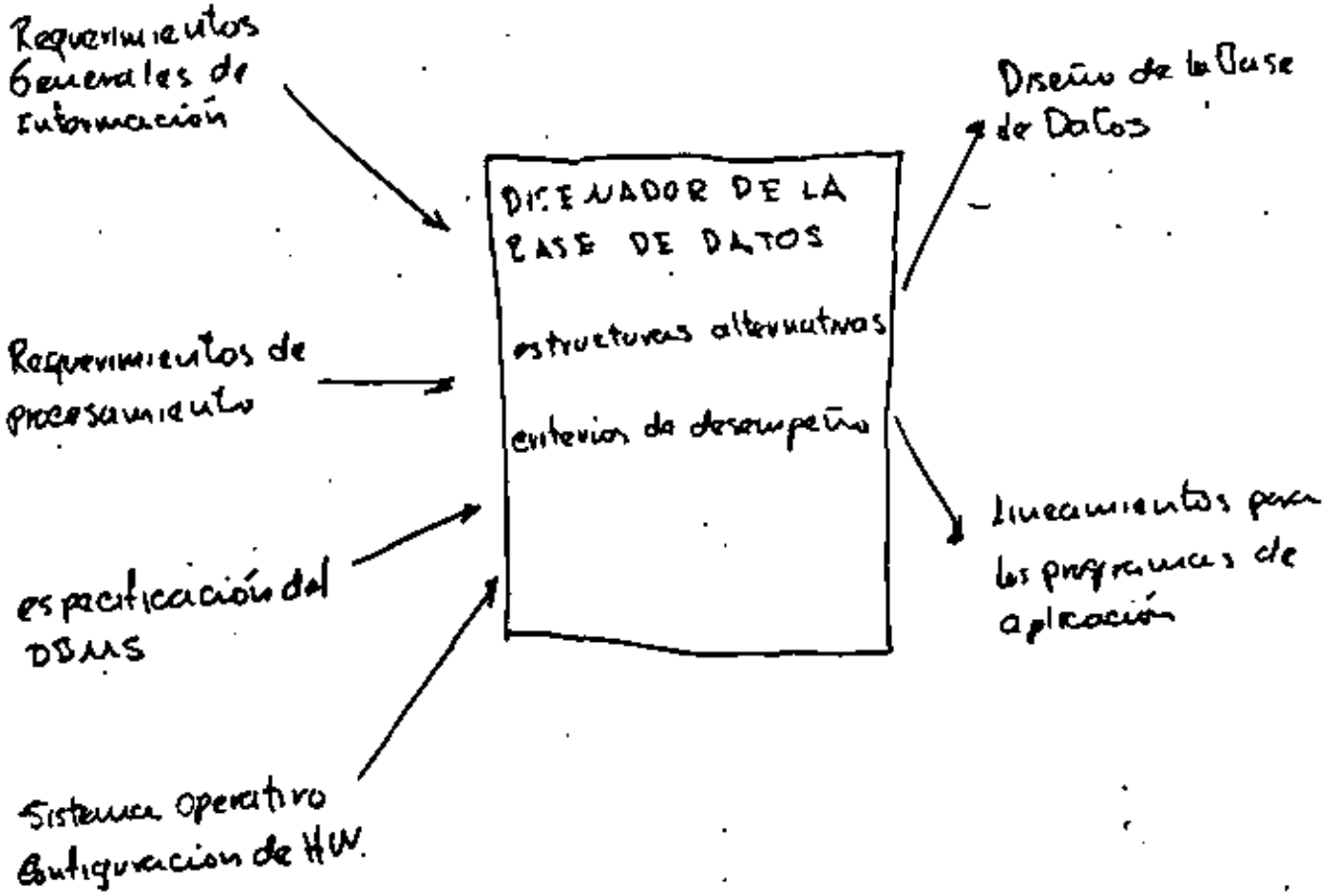
DISEÑO DE BASES DE DATOS

APUNTES ADICIONALES

ING. DANIEL RIOS ZERTUCHE

JUNIO, 1984

PROCESO DE DISEÑO DE LA BASE DE DATOS



Revisiones del Diseño

Despues del Diseño Conceptual
Despues del Diseño Detallado
Despues de la implantación
Despues de entrar en producción

F1

Ciclo de Vida de un Sistema

Definición del problema
Estudio de factibilidad
Analisis
Diseño del Sistema
Diseño detallado
Implantación
Mantenimiento

CICLO DE VIDA DE UN SISTEMA DE BASE DE DATOS

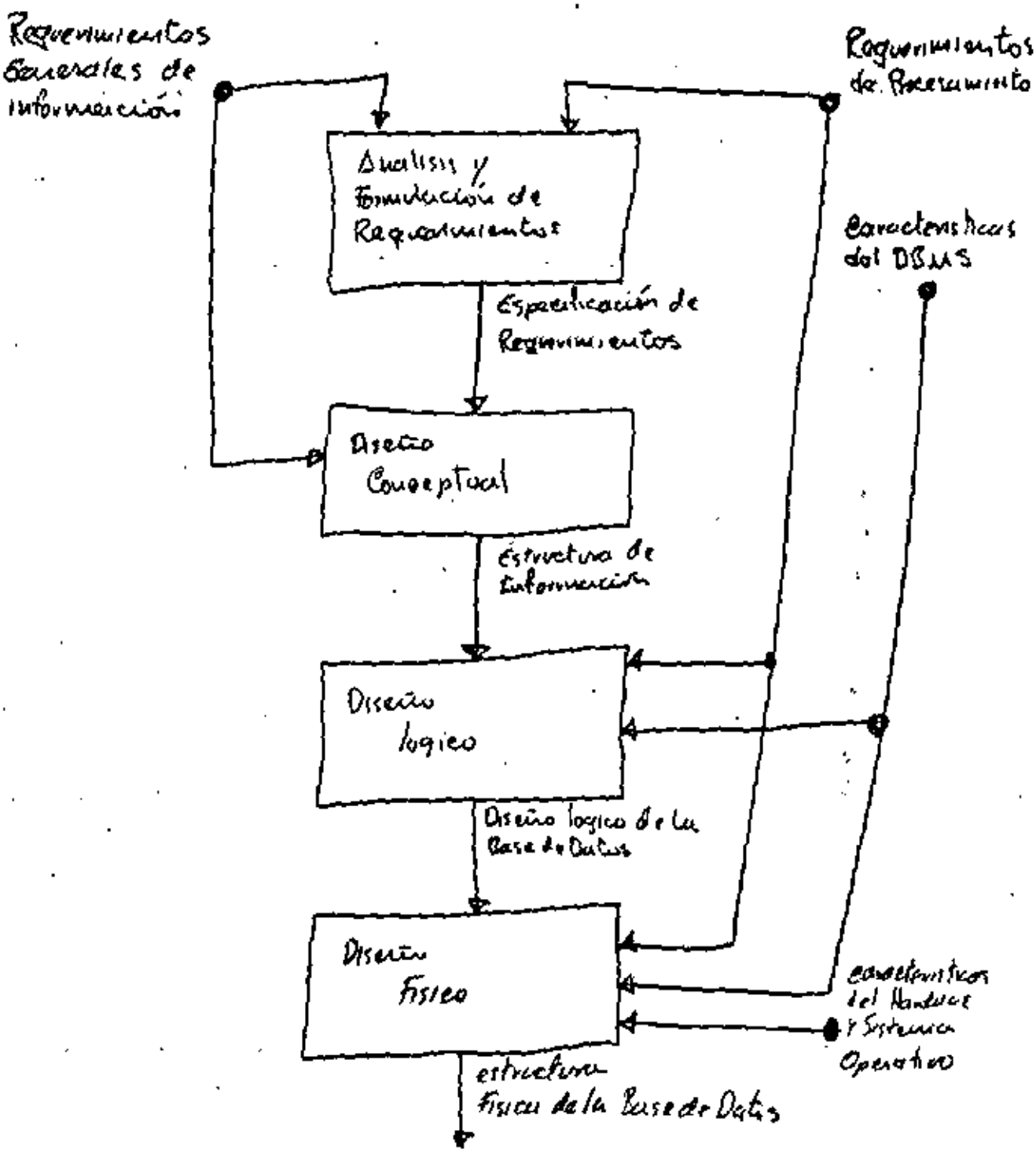
Fase de Analisis y Diseño

1.- Analisis y Formulación de Requerimientos
2.- Diseño Conceptual
3.- Diseño de la implantación
4.- Diseño Fisico

Fase de implantación y Operación de la Base de Datos

1. Implantación de la Base de Datos
2.- Operación y Monitoreo
3.- Modificación y Adaptación

PASOS BASICOS EN EL DISEÑO DE UNA BASE DE DATOS



INTEGRIDAD

+ Restricciones
+ Reglas para imponer las restricciones
+ Reglas para tratar a los datos cuando fallan en cumplir las reglas de consistencia
+ imposición eficiente de las restricciones

CONSISTENCIA

+ Metodos para modificar Datos
++ imponer tiempo de espera antes de modificar
++ imponer tiempo de espera a los demas usuarios activos
++ El dominio de influencia de la modificación
++ El algoritmo para la modificación de los Datos
+ Numero de usuarios de la Base de Datos en el instante que se lleva a cabo la modificación

ANÁLISIS Y FORMULACIÓN DE REQUERIMIENTOS F5

1. Identificación de las organizaciones que desempeñan las funciones operacionales

2. Solicitar a cada encargado la siguiente información
   Puestos de las personas en su área de responsabilidad
   funciones operativas desempeñadas por puesto
   Objetivo que cubre cada puesto

3. Clasifique los puestos en planeación
   control
   operación

4. entreviste a dos personas en cada puesto operacional

5. entrevistas a los ~~funciones~~ gerentes respecto a las funciones de Planeación y Control

Entrevistas Operativas

Objetivos:

+ Identificar cada función Operacional
+ Identificar sus datos
+ Identificar las reglas implícitas y explícitas de como y cuando cada función ocurre

Procedimiento para la entrevista

1.- para frecuencia = diaria, semanal, mensual, Trimestral y anual

pida al entrevistado describir
funciones o tareas

Documente en un diagrama de Flujo
acciones principales
decisiones
interfases.

Utilice el diagrama como un mecanismo para realimentación

2º Una vez corroborados los diagramas determine

documentos
archivos
referencias informales

3º Determine la relación entre el diagrama de flujo y los referencias. De acuerdo con el entrevistado

4º Solicite una muestra de cada documento

5º Asocie cada elemento dato en los documentos con el símbolo en el diagrama de flujo en el que sea utilizado.

6º anexe a cada Diagrama de Flujo una nota describiendo las reglas.

ENTREVISTA A GERENCIAS

ENTREVISTA A NIVELES MEDIOS

Interfase entre las areas operativas

Reglas que gobiernas las operaciones diarias

Información requerida para medición y control del desempeño

Efectos potenciales de cambios pronosticados en las areas operacionales.

Transformación de los Requerimientos de Información

Identificación de los elementos Data

organice los elementos data a listas genericas
resuelva redundancia y
definición

Identifique las tareas operacionales

tarea: una unidad unica de trabajo

Consiste de una secuencia de pasos dirigidos hacia una meta comun.

todos los pasos crean o usan los mismos datos.

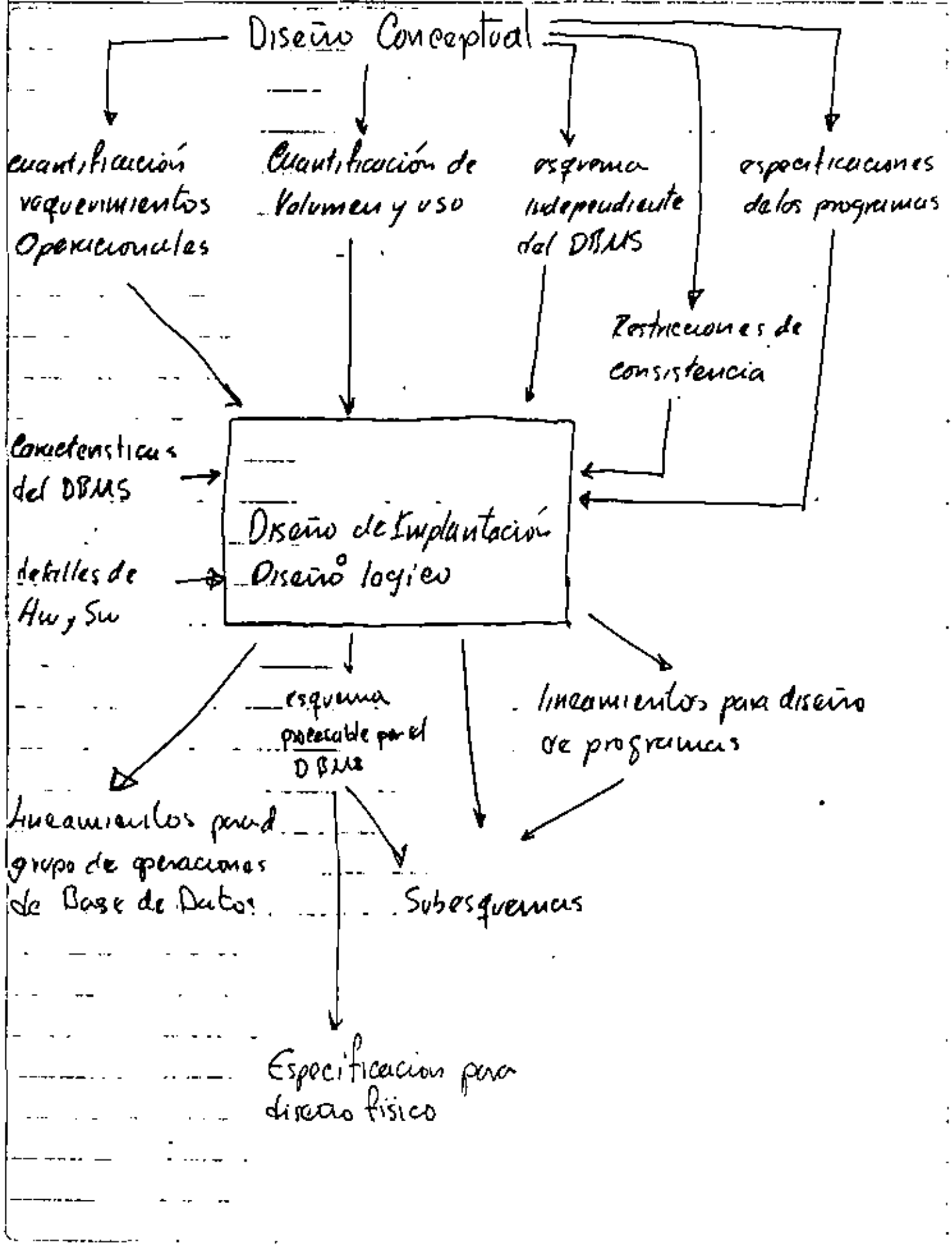
Compare las entrevistas repetidas

Documente

+ asigne un identificador de tarea
+ defina la tarea a través de un verbo-objeto
+ clasifique las tareas como operacionales de control o planeación
+ la frecuencia de la tarea
+ relacione cada tarea con el proceso de que es parte

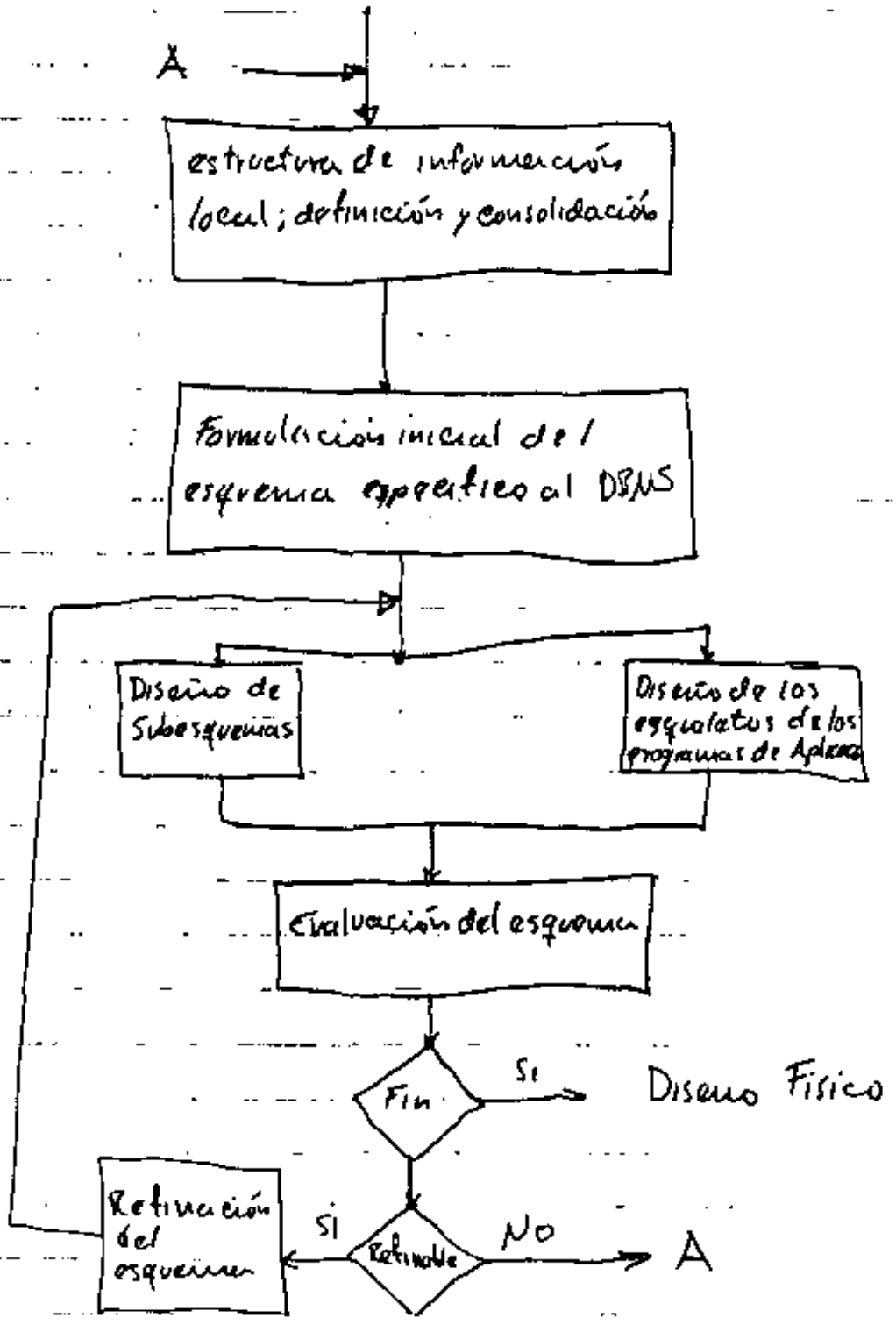
Identifique las tareas de Control y Operacionales

Identifique las políticas operativas actuales y futuras.


Diseño Conceptual
cuantificación requerimientos Operacionales
Cuantificación de Volumen y uso
esquema independiente del DBMS
especificaciones de los programas
Restricciones de consistencia
Características del DBMS
detalles de Hw y Sw
Diseño de Implantación
Diseño lógico
esquema procesable por el DBMS
lineamientos para diseño de programas
lineamientos para grupo de operaciones de Base de Datos
Subesquemas
Especificacion para diseño físico

Pasos del Diseño

esquema Conceptual



Desempeño de la Estructura logica de la Base de Datos

Accesos a registros logicos

$$LRA_i = \sum_{j=1}^{N} LRA_{ij}$$

Aplicación i
numero de tipos de Registro N

Accesos por unidad de tiempo

$$LRA = \sum_{i=1}^{M} \sum_{j=1}^{N} LRA_{ij} \times F_i$$

M = numero total de aplicaciones
F = frecuencia de ejecución de la aplicación

Volumen de Transporte

$$TRVOL_i = \sum_{j=1}^{N} LRA_{ij} \times RECSIZE_j$$

Volumen de Transporte por unidad de tiempo

$$TRVOL = \sum_{i=1}^{M} \sum_{j=1}^{N} LRA_{ij} \times RECSIZE \times F_i$$

Espacio de Almacenamiento

$$DSTOR = \sum_{j=1}^{N} RECSIZE_j \times NREC_j$$

Almacenamiento de Apuntadores

$$PTRSTOR = \sum_{j=1}^{N} NREC_j \times PS \times NPTR_j$$

PS = Tamaño del apuntador en bytes
$NPTR_j$ = numero promedio de apuntadores almacenados con registro tipo j

1.1 ARCHIVOS

Archivo

Un archivo se define como una colección de registros similares contenidos en los dispositivos de almacenamiento secundario de la computadora.

Registro

Una colección de campos relacionados conteniendo objetos-dato elementales.

Objeto dato

Típicamente es representado por un valor el cual es parte de la descripción de un objeto o un ~~escrito~~ evento.

1.1.1 Tamaño de la Base de Datos

Grande. Implica una cantidad mayor de la que una persona pueda manejar por si misma aún con la ayuda de la computadora. Actualmente la cantidad puede variar dependiendo de la complejidad de la - Aplicación.

Muy grande. Una definición formal puede ser una base de datos tan grande que el tiempo necesario para hacer una copia es mayor que el intervalo entre -

las transacciones necesarias para mantener la Base de Datos actualizada.

1.1.2 Organización de Archivos

1.1.3 Entrada-Salida

1.2 Computaciones en la Base de Datos

Siguiendo la terminología de Tsichritzis

El termino computación se usa para denotar una sección de una aplicación la cual manipula la Base de Datos. La mayoría de las computaciones las cuales son usadas para manipular una colección de datos son conceptualmente simples.

Reconocemos cuatro clases de computaciones relacionadas con las Bases de Datos.

1. La Construcción de la Colección de Datos
2. La Actualización de los elementos dado en una colección de datos.
3. La Recuperación de Datos de una colección de Datos.
4. La Reducción de gran cantidad de Datos a una forma usable.

1.2.1 Procesos

El Sistema Operativo el cual organiza ~~el itinerario para~~ la -

ejecución de la computación solicitada por el usuario pue de descomponer la computación en un número de procesos distintos.

Proceso. Es la unidad computacional básica que maneja el siste ma operativo.

1.2.2 Secciones de Procesos

Un proceso puede ser descompuesto en un número de sec ciones.

Sección. Una sección es una colección de pasos de un pro grama durante los cuales el sistema operativo no se tiene que preocupar acerca de el proceso.

Una sección del proceso la cual presenta interacciones poten ciales es conocida como una sección crítica.

1.3 Una visión Jerárquica de los Datos

Conceptos
Información
Hechos y Descripción — Sujeto de
Datos Estructurados — Estudio
Representaciones de Datos
Bits y Bytes Estructurados
Hardware de Almacenamiento bits y bytes

1.4 Practica Actual

1.4.1 Un programa de archivos

1.4.2 Control del Sistema Operativo

1.5 Descripciones

En orden para permitir la implantación y uso de un sistema por un número de personas alguna documentación debe existir para describir tanto el aspecto estático, la estructura de la Base de Datos como el aspecto procedural, las computaciones las cuales llevan a cabo las transformaciones de Datos.

Una fotografía de los datos en terminos de archivos, registros, campos y la relación entre los elementos dato contenidos en estos elementos es apropiado para el aspecto estático.

La descripción procedural puede ser dada como una fórmula, una descripción de secciones de programa a ser ejecutadas, o un diagrama de flujo. Debido a los problemas de comunicación que se presentan en los equipos de programación y que a la fecha no hay métodos formales para verificar completamente la corrección para procesos de inte-

racción. Nosotros por tanto preferimos descripciones detalladas de los datos estáticos para el desarrollo de Bases de Datos.

Descripciones Estáticas contra Descripciones Procedurales

Los aspectos procedurales y estáticos frecuentemente ocurren juntos cuando describimos archivos. Para muchos problemas sencillos de procesamiento de Datos, sin embargo no es necesario ninguna descripción de proceso. En aquellos casos en que el proceso de computación es descrito indirectamente proveyendo dos fotografías del archivo, especificando la forma y contenido antes y después de la computación. Este método también es aplicable a el movimiento de datos entre dos archivos o bases de datos separadas. El programador usa las Tablas de antes y después para escribir el código. Prescindir de la codificación por completo en la intensión de los generadores de reportes, los cuales aceptan la definición de los archivos de entrada y la descripción de los reportes de salida.

En otros casos uno puede encontrar que la organización de los datos esta implícitamente definida por un listado de los pasos de el proceso que pone los datos en la Base de Datos. Mucho

análisis es necesario cuando nuevos programas desean acceder los datos que solo han sido definidos proceduralmente. Las dos elecciones accesibles están implícitas en la existencia de dos terminos para la descripción de las estructuras de almacenamiento: organización de archivos frente a métodos de acceso.

Elementos dato especiales, descritos y mantenidos en la Base de Datos pueden ser usados para controlar la dinámica del programa. Un elemento dato de una Base de Datos puede establecer que un archivo particular han sido completamente actualizado por una computación. Tal información puede ser usada para garantizar la correcta secuencia de ejecución de programas que comparten datos. Los elementos datos mantenidos en cada registro pueden especificar que ciertos registros aún no han sido actualizados, posiblemente por la falta de datos de entrada, tal como las horas trabajadas por algunos empleados.

Tales computaciones pueden ser programadas para decidir no proseguir en tal caso; otros procesos no deben ser impedidos a menos que ellos requieran uno de los registros que no se han actualizado. Nos referimos a los datos que no se han me

tido a la Base de Datos pero que describen el estado de la Base de Datos para el control de las computaciones como datos de control.

Vemos que hay una relación entre la calidad o lo completa de nuestra descripción de los datos y nuestra habilidad para especificar las secciones de el programa los cuales se combinan para formar computaciones de proceso de datos.

1.6 Ligado

Tiempo de Ligado

1.7 Clasificación de Sistemas Operativos

Los sistemas operativos proveen una gran variedad de servicios los cuales los principales pueden clasificarse como:

Secuenciamiento de Procesos

Soporte del Sistema de Archivos

Soporte de Entrada-Salida

Medición de Uso

Los Sistemas de archivos incluyen programas para mantener directorios de archivos y garantizar su seguridad a largo - plazo y programas invocados por el usuario para accesar,

esto es, leer y escribir los archivos durante las tareas computacionales.

1.7.1 Procesamiento Batch

1.7.2 Multiprogramación

1.7.3 Procesamiento de Transacciones

El enfoque más importante de Sistema Operativo para Base de Datos Interactivos es llamado procesamiento de transacciones. Este método asume que las computaciones son naturalmente pequeñas en terminos de espacio y tiempo. El Software del sistema de Base de Datos se asume esta siempre accesible, de manera que las computaciones de los usuarios consisten principalmente de llamadas a la Base de Datos para lectura y escritura, y una cantidad limitada de computación lógica y numérica. Tal computación es llamada una transacción. Las transacciones son iniciadas tan rápido como es posible después que el usuario la solicita y se les permite correr tanto como sea necesario para cumplir los requerimientos computacionales.

Una transacción es también formalmente definida como una computación la cual, después de completarse, deja la Base

de Datos en un estado bien definido, o al menos en un estado no peor que antes. Una falla durante la transacción requiere que la Base de Datos sea restaurada a su estado original. Si únicamente transacciones bien comportadas operan en la Base de Datos, la Integridad de la Base de Datos siempre es mantenida.

En un medio ambiente orientado a transacciones generalmente se ejerce un control cercano sobre las computaciones para prevenir que una computación defectuosa atore todo el sistema.

1.8 Aplicaciones

1.9 Revisión

Organización de un Sistema de Archivos Básico

3.0 Introducción

Las características básicas deseadas de sistemas que almacenan grandes cantidades de datos son:

+ Acceso rápido para recuperación
+ Actualización conveniente ~~conocimiento~~
+ Economía de almacenamiento

Criterios Secundarios importantes son:

+ Capacidad para representar estructuras de información del mundo real.
+ Confiabilidad
+ Protección de privacidad
+ Mantenimiento de integridad

El Diseño de Bases de Datos requiere análisis para la predicción del desempeño y esto requiere que la organización de los archivos sea fácilmente abstraída. Todos estos criterios tienden a ser conflictivos entre sí. La elección del método de organización de archivos determina la relativa

conveniencia de un sistema de acuerdo con todos estos criterios.

El número de alternativas posibles en la organización de archivos es practicamente ilimitado. En orden de hacer el proceso de selección manejable, describiremos y mediremos - seis alternativas básicas de diseño de archivos. La mayoría de las estructuras usadas en la practica ya sea o caen en una de estas categorías o puede reducirse a una combinación de estos métodos. Los métodos seleccionados son también - razonables en terminos de los criterios secundarios.

Los seis tipos básicos son:

Pila

Secuencia

Secuencial Indexado

Indexado

Directo

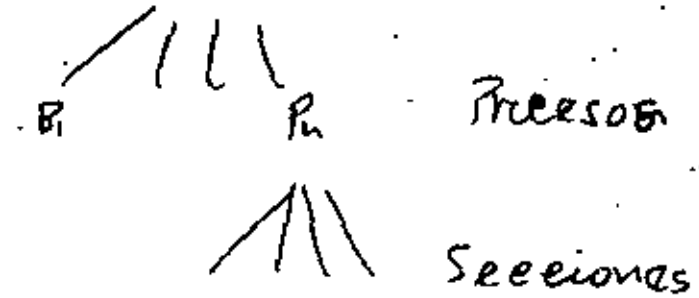
Multianillo

Expandiremos la definición previa de archivo a incluir que no solo es un conjunto consistente en registros similares sino - también tiene una organización consistente.

Computaciones

1: Construcción de una colección de datos
2. Actualización de los elementos en una colección de datos
3. La recuperación de datos de una colección de datos
4. La Reducción de datos en gran cantidad a una forma usable

Computacion



Servicios del sistema operativo

- Secuenciamiento de procesos
- Soporte del sistema de archivos
- Soporte a entrada salida
- Medición de uso

Tipos de Sistemas operativos

Batch
Multiprogramación
Tiempo compartido
Procesamiento de transacciones

Organización de un Sistema de Archivos

Características Básicas deseadas

+ Acceso rápido para recuperación
+ Actualización conveniente
+ Economía de almacenamiento

Características Secundarias

+ Capacidad para representar estructuras de información del mundo real
+ Confiabilidad
+ Protección de Privacidad
+ Mantenimiento de la Integridad

Parametros dependientes del Hw

Tamaño del Bloque B

Tiempo de Seek s

latencia rotacional r

Capacidad

	Tamaño del Bloque B				
	1	80	200	1000	3000
Disco 3330	730 k	37.1 M	68.4 M	83.6 M	91.2 M
Disco 3380	79.6 M	2119.7 M	2272 M	2442 M	2389 M

Factor de Bloque

$$Bfr = \left\lfloor \frac{B}{R} \right\rfloor \quad \text{fijo}$$

B = longitud del Bloque

R = longitud del registro

$$Bfr = \frac{B-P}{R+P} \quad \text{variable spanned}$$

$$Bfr = \frac{B-\frac{1}{2}R}{R+P} \quad \text{variable unspanned}$$

P = tamaño del apuntador

Desperdicio

$$W_G = G/Bfr$$

$$W = W_G + W_R$$

$$W \approx G/Bfr \quad \text{fijo}$$

$$W = P+(P+G)/Bfr \quad \text{var-spanned}$$

$$W = P + \frac{\frac{1}{2}R+G}{Bfr} \quad \text{var-unspanned}$$

Tasa de transferencia t

Tiempo para leer un registro de R caracteres

$$T_R = \frac{R}{t} \quad \text{ms.}$$

Tiempo requerido para leer un bloque completo

$$b_{tt} = \frac{B}{t}$$

tasa de transferencia efectiva

Efecto de los Gaps y bloqueaje en la tasa de transferencia

$$t_{ayl} = t \frac{R}{R+W} \quad \text{car/ms}$$

Efecto de los seeks en la tasa de transferencia

$$s' = s / B_{tr}$$

$$2r = n r t (R+W) / t$$

$$s'_{mínimo} = \frac{2r}{k\,n r t} = \frac{1}{k} \frac{R+W}{t}$$

$$\frac{1}{k} \frac{R+W}{t} \leq s' \leq \frac{s}{B_{tr}}$$

estimado

$$s' \approx \frac{2r}{n r t} = \frac{R+W}{t}$$

(12)

$$t' = \frac{R}{(R+W)/t + s'} \quad \text{chars/ms}$$

Empleo de la tasa de transferencia anterior

Acceso secuencial a registros

use t' aplicada a R

Acceso secuencial a bloques

use t' y los datos x bloque $Bf_r \times R$

Densidad

Localidad

Fuerte

↕

debil

Actualización de Bloques

$T_{RW} = 2r$ + Contabilización $<< 2r$

Descripción de archivos

R : Cantidad de almacenamiento requerida por registro

T_F : tiempo necesario para traer un registro arbitrario

T_N : tiempo necesario para obtener el siguiente registro

T_I : tiempo para actualizar un archivo insertando un registro

T_U : tiempo para actualizar un archivo cambiando un registro

T_X : tiempo necesario para lectura exhaustiva del archivo

T_Y : tiempo necesario para reorganizar el archivo

Desempeño de / archivo Pila

a = número total de tipos de atributos en el archivo.
a' = número promedio de atributos por registro
A = longitud promedio de la descripción de un atributo
V = longitud promedio de la porción del valor

Tamaño del Registro

$$R = a'(A + V + 2)$$

Trae un registro en una pila

b = número de bloques en el archivo

promedio de bloques leídos $= \frac{1}{2} b$ si $b \gg 1$

tiempo de E/S para leer este número de bloques secuencialm
usando transferencia masiva

$$T_F = \frac{1}{2} b \frac{B}{t'}$$

ya que $bB = nR$

$$T_F = \frac{1}{2} n \frac{R}{t'}$$

Obtén el siguiente registro de una pila

$$T_N = T_F$$

Inserta en una pila

$$T_I = S + r + bff + T_{RW}$$

haciendo un lote de agregaciones simplificamos a

$$T_I = S + 3r + bff$$

Actualiza registros en una Pila

$$T_U = T_F + T_{RW} + T_I$$

Elimina un registro

$$T_D = T_F + T_{RW}$$

Lee la pila completa

$$T_X = 2\,T_F = n\frac{R}{f}$$

Reorganización de una Pila

$$T_Y = (u+o)\frac{R}{\epsilon'} + (u+o-d)\frac{R}{\epsilon'}$$

aquí

$$o = n_{insert} + M$$

o = número de registros agregados
n_{insert} = número de registros insertados
M = número de registros actualizados

$$d = n_{delete} + \mu$$

d = número de registros movidos para eliminación
n_{delete} = número de Registros eliminados
μ = número de registros que fueron actualizados

Desempeño del Archivo Secuencial

Tamaño de Registro

$$R = a\,V$$

Trae un Registro

$$T_F = \frac{1}{2}\, n\, \frac{R}{t'}$$

Obten el siguiente registro

$$T_N = \frac{bft}{Bfr} = \frac{R}{t'}$$

Inserta en un archivo Secuencial

Actualiza un archivo Secuencial

Lee el archivo Completo

$$T_X = [\text{illegible}]\, n\, \frac{R}{t'}$$

Reorganización

1. Orden

Número de Orden
Clave de Vendedor
Nombre del Vendedor
No. de Cliente
Nombre del Cliente
Fecha Orden
Clave del que toma la Orden
Cantidad Ordenada
Número de Referencia
Descripción del Producto

2. Catálogo de Productos

Número de Referencia
Descripción del Producto

3. Llamada a Bodega

Número de Referencia
Cantidad Ordenada
Hay Existencia
Fecha esperada por el Embarque
Precio Unitario

4. Llamada a Vendedor

Prod. que no hay Existencia
Núm. de Ref. del Prod. Alternativo
Descripción del Prod. Alternativo

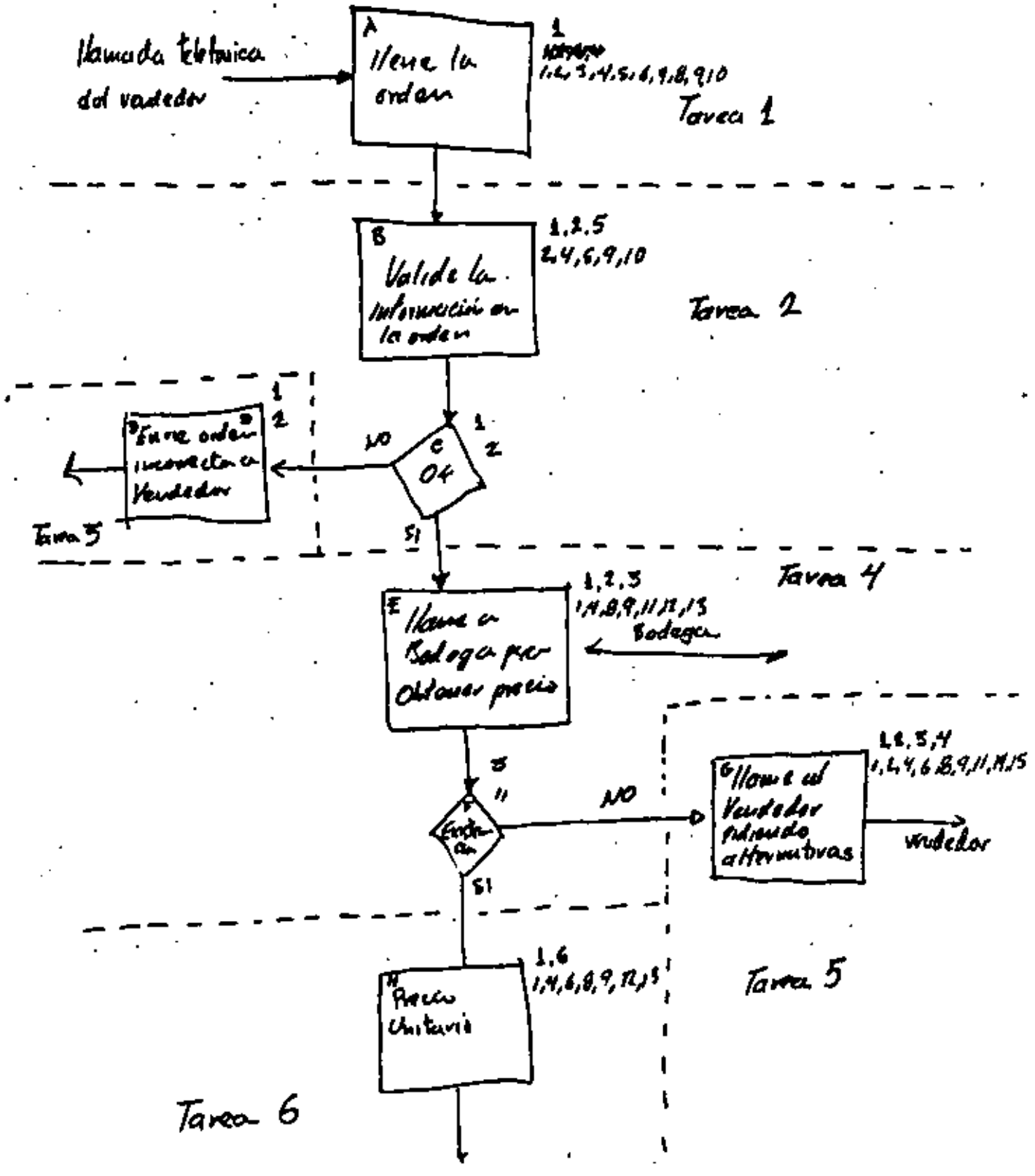
5. Kardex del Cliente

No. de Cliente
Nombre del Cliente
Clave del Vendedor

6. Orden de Precio

Número de Orden
Fecha
Cantidad Ordenada
Número de Referencia
Precio Unitario
Fecha de Embarque

		1	2	3	4	5	6
1.	Número de Orden	X					X
2.	Clave del Vendedor	X				X	
3.	Nombre del Vendedor	X					
4.	No. de Cliente	X				X	
5.	Nombre del Cliente	X				X	
6.	Fecha Orden	X					X
7.	Clave del que tomar Orden	X					
8.	Cantidad Ordenada	X		X			X
9.	Número de Referencia	X	X	X	X		X
10.	Descripción del Prod.	X	X				
11.	Hay Existencia			X			
12.	Fecha para Embarque			X			X
13.	Precio Unitario			X			X
14.	Prod. Alternativo				X		
15.	Descr.Prod. Alternativo				X		


Llamada telefonica del vendedor
A
Llene la orden
1
1,2,3,4,5,6,7,8,9,10
Tarea 1
B
Valide la información en la orden
1,2,5
2,4,6,9,10
Tarea 2
C
OK
1
2
NO
D
Envíe orden incorrecta a Vendedor
1
2
Tarea 3
SI
Tarea 4
E
Llame a Bodega por Obtener precio
1,2,3
1,7,8,9,11,12,13
Bodega
F
Orden OK
8
11
NO
G
Llame al Vendedor pidiendo alternativas
1,2,3,4
1,4,7,6,8,9,11,12,13
vendedor
SI
H
Precio Unitario
1,6
1,7,6,8,9,12,13
Tarea 5
Tarea 6

TAREA	ELEMENTOS DATO
1	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
2	2, 4, 5, 9, 10
3	2
4	1, 4, 8, 9, 11, 12, 13
5	1, 2, 4, 6, 8, 9, 11, 14, 15
6	1, 4, 6, 8, 9, 12, 13

MATRIZ DE VINCULOS

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1		2	1	3	1	2	1	3	3	1	2	1	1	1	1
2	2		1	3	2	2	1	2	3	2	1			1	1
3	1	1		1	1	1	1	1	1	1					
4	3	3	1		2	2	1	3	4	2	2	1	1	1	1
5	1	2	1	2		1	1	1	2	2					
6	2	2	1	2	1		1	2	2	1	1			1	1
7	1	1	1	1	1	1		1	1	1					
8	3	2	1	3	1	2	1		3	1	2	1	1	1	1
9	3	3	1	4	2	2	1	3		2	2	1	1	1	1
10	1	2	1	2	2	1	1	1	2						
11	2	1		2		1		2	2			1	1	1	1
12	1			1				1	1		1		1		
13	1			1				1	1		1	1			
14	1	1		1		1		1	1		1				1
15	1	1		1		1		1	1		1		1		

VECTOR DE CUENTA DE USO POR TAREA

1	4
2	4
3	1
4	5
5	2
6	3
7	1
8	4
9	5
10	2
11	2
12	2
13	2
14	1
15	1

VECTOR DE TOTAL DE VINCULOS

1	14
2	12
3	9
4	14
5	9
6	12
7	9
8	14
9	14
10	9
11	10
12	6
13	6
14	8
15	8

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1		.5	.25	.75	.25	.5	.25	.75	.75	.25	.5	.25	.25	.25	.25
2	.5		.25	.75	.5	.5	.25	.5	.75	.5	.25			.25	.25
3	1	1		1	1	1	1	1	1	1					
4	.6	.6	.2		.4	.4	.2	.6	.8	.4	.4	.2	.2	.2	.2
5	.5	1	.5	.5		.5	.5	.5	1	1					
6	.66	.66	.33	.66	.33		.33	.66	.66	.33	.33			.33	.33
7	1	1	1	1	1	1		1	1	1					
8	.75	.5	.25	.75	.25	.5	.25		.75	.25	.5	.25	.25	.25	.25
9	.6	.6	.2	.8	.4	.4	.2	.6		.4	.4	.2	.2	.2	.2
10	.5	1	.5	1	1	.5	.5	.5	1						
11	1	.5		1		.5		1	1			.5	.5	.5	.5
12	.5			.5				.5	.5		.5		.5		
13	.5			.5				.5	.5		.5	.5			
14	1	1		1		1		1	1		1				1
15	1	1		1		1		1	1		1			1	

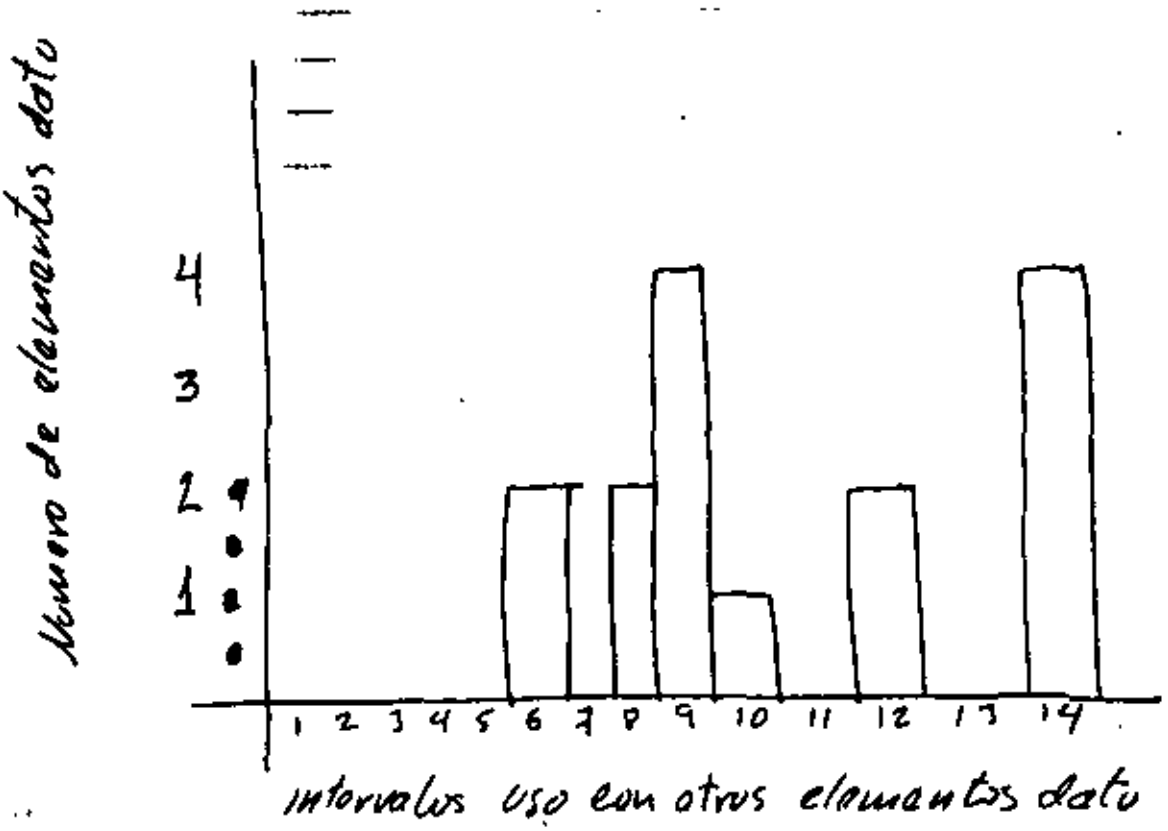
vector de alto uso .70

1	3
2	2
3	9
4	1
5	3
6	0
7	9
8	3
9	1
10	4
11	4
12	0
13	0
14	8
15	8

vector de radio de uso

1	21.42
2	16.66
3	100.00
4	7.14
5	33.33
6	0.0
7	100.00
8	21.42
9	7.14
10	44.44
11	40.00
12	0.0
13	0.0
14	100.00
15	100.00

HISTOGRAMA USO CON OTROS ELEMENTOS DATO



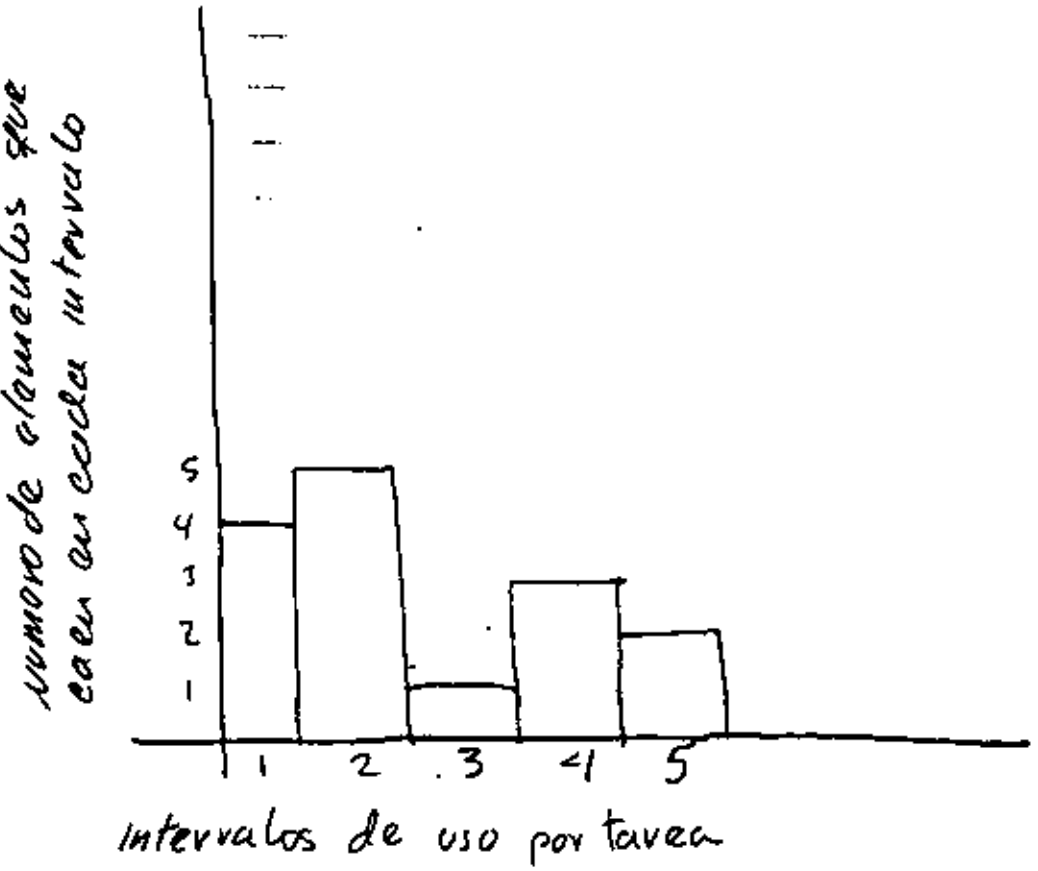
VECTOR DE TOTAL DE VINCULOS

1	14
2	12
3	9
4	14
5	9
6	12
7	9
8	14
9	14
10	9
11	10
12	6
13	6
14	8
15	8

FRECUENCIAS

0	
1	
2	
3	
4	
5	
6	2
7	
8	2
9	4
10	1
11	0
12	2
13	0
14	4

HISTOGRAMA USO A TRAVEZ DE TAREAS



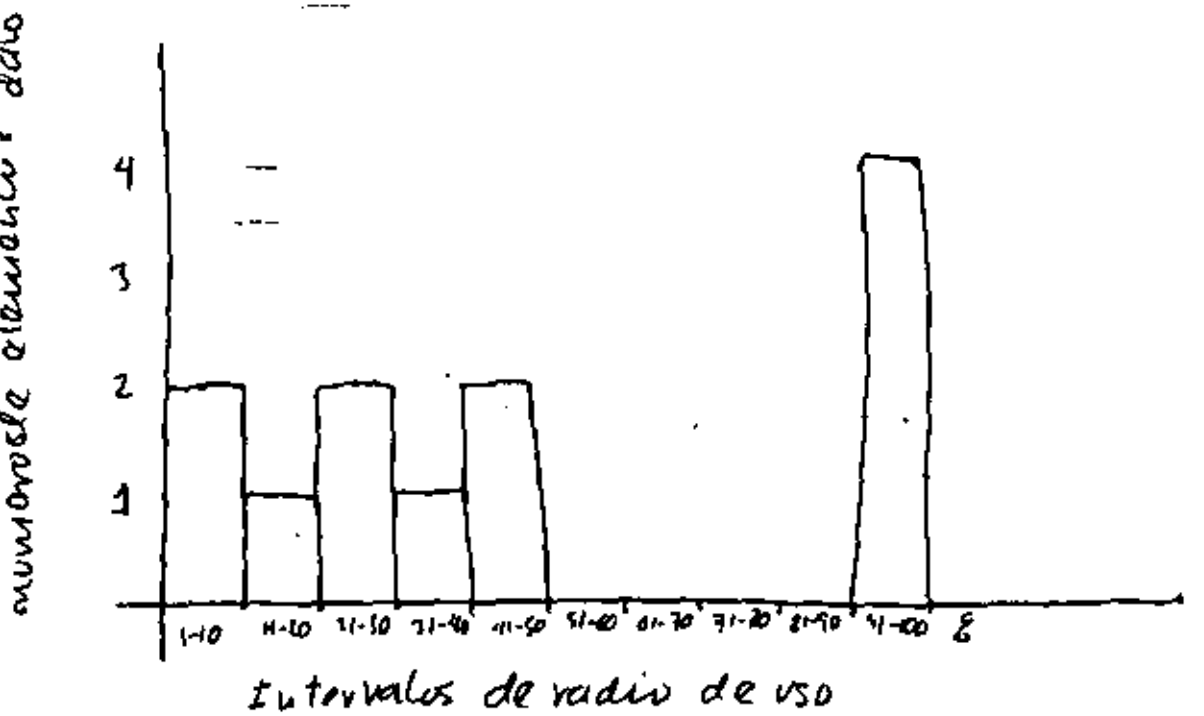
VECTOR DE CUENTA DE USO POR TAREA

1	4
2	4
3	1
4	5
5	2
6	3
7	1
8	4
9	5
10	2
11	2
12	2
13	2
14	1
15	1

FRECUENCIAS

0	0
1	4
2	5
3	1
4	3
5	2

HISTOGRAMA RADIO DE USO



VECTOR DE RADIO DE USO

1	21.42
2	16.66
3	100.0
4	7.14
5	33.33
6	0.0
7	100.0
8	21.42
9	7.14
10	44.44
11	40.0
12	0.0
13	0.0
14	100.0
15	100.0

FRECUENCIA

1-10	2
11-20	1
21-30	2
31-40	1
41-50	2
51-60	
61-70	
71-80	
81-90	
91-100	4

	USO A TRAVES DE ___ LAS TAREAS	USO JUNTO CON OTROS ELEMENTOS DATO	RADIO DE USO
ALTO			
PROM			
BAJO			

	USO A TRAVES	USO CON OTROS DATOS	RADIO DE USO
ENTIDAD	ALTO	ALTO	BAJO
ATRIBUTO REFERENCIADO POR ENTIDAD NO UNICA	PROM	PROM	PROM
ATRIBUTO REFERENCIADO POR ENTIDAD UNICA	BAJO	BAJO	ALTO

Elemento dato	Uso a través de las tareas	Uso con otros datos	Razón de uso	Observ.
1	4	14	21	Entidad
2	4	12	16	Entidad
3	1	9	100	
4	5	14	7	Entidad
5	2	9	33	
6	3	12	0	—
7	1	9	100	
8	4	14	21	Entidad
9	5	14	7	Entidad
10	2	9	44	
11	2	10	40	
12	2	6	0	—
13	2	6	0	—
14	1	8	100	
15	1	8	100	

ENTIDADES POR TAREA

TAREA	ENTIDADES
1	1, 2, 4, 8, 9
2	2, 4, 9
3	2
4	1, 4, 8, 9
5	1, 2, 4, 8, 9
6	1, 4, 8, 9

Candidatos a antidiabéticos no micos	número de apariciones juntos	Frecuencia
1, 2	2	33
1, 4	4	60
1, 8	4	60
1, 9	4	60
2, 4	2	33
2, 8	2	33
2, 9	3	50
4, 8	4	60
4, 9	5	83
1, 2, 4	2	33
1, 2, 8	2	33
1, 2, 9	2	33
1, 2, 4, 8	2	33
1, 2, 4, 9	2	33
1, 2, 7, 8, 9	2	33

Elementos dato usados por candidatos a Entidades no unicas

Candidatos	Tareas en que aparece	Otros elementos dd
1, 4	1	3, 5, 6, 7, 10
	4	11, 12, 13
	5	6, 11, 14, 15
	6	6, 12, 13
1, 8	1	3, 5, 6, 7, 10
	4	11, 12, 13
	5	6, 11, 14, 15
	6	6, 12, 13
1, 9	1	3, 5, 6, 7, 10
	4	11, 12, 3
	5	6, 11, 14, 15
	6	6, 12, 13
4, 8	1	3, 5, 6, 7, 10
	4	11, 12, 13
	5	6, 11, 14, 15
	6	6, 12, 13
4, 9	1	3, 5, 6, 7, 10
	2	5, 10
	4	11, 12, 13
	5	6, 11, 14, 15
	6	6, 12, 13

Identificación de Vinculos

Politicas

1:- Un cliente es asignado a un vendedor

2:- el que toma la llamada es responsable del proceso de la orden

3:- Se aceptan ordenes solo del vendedor que maneja a el cliente

Directorio de Alumnos del curso: DISEÑO DE BASES DE DATOS

Del 14 al 28 de Mayo

1984

1. Héctor Alvarez Camacho
General Electric de México, S.A. de C.V.
Vía Morelos 351 Km.17.5 Carr.a Laredo
Cerro Gordo,Edo. de México 55500
569 00 16

Av.Tamaulipas 190-302
Condesa
Cuauhtémoc
México, D.F.
515 02 13

2. Antonio Aranda Pastor
PEMEX
Marina Nal. 436
V.Anzures
México, D.F.

Sánchez Azcona 1651-801
Valle
B.Juárez
03100 México, D.F.
688 68 48

3. Jorge O. Arce Marín
C. F. E.
M. Alemán 91
Chilpancingo, Gro.
2 21 36

Rancho del Arco 76-103
Los Girasoles
Coyoacán
04920 México, D.F.
581 60 75

4. Francisco Barrera Zárate
Fac. de Est. Sup. Cuautitlán
Rancho Almaraz
Cuautitlán Izcalli
Edo.de México

Av. Chapultepec 513-9
Juárez
Cuauhtémoc
06600 México, D.F.
553 82 91

5. Clara S.Bermudez Ruiz
Shakespeare 192-8
Anzures
M.Hgo.
11590 México, D.F.
545 50 10

6. Rodolfo Bibiano Nava
PEMEX
Marina Nal. 239
México, D.F.
254 20 44

Hidalgo 39 Edif. B 405
Atzcapotzalco
02000 México, D.F.
352 38 35

7. Gilles Boud'hors Leautaud
Papmas 805-2-9-1
Lomas
M.Hgo.
11000 México, D.F.
540 36 66

8. Ignacio Carbajal Carbajal
IMP
Eje Lázaro Cárdenas 152
G.A.Madero
07730 México, D.F.
567 66 00

Fernando Ramírez 53
Obrera
Cuauhtémoc
06800 México,D.F.
578 52 45

9. Dimas Contreras Figueroa
PEMEX
Marina Nal. 329
Anzures
Cuauhtémoc
México, D.F.
254 38 14

Edif. 70 Ent. B Depto.302
U.Cuitláhuac
Cosmopolita
Azcapotzalco
254 38 14

10. Joaquín Cortés Romero
DECFI
Tacuba 5
Centro
06000 México, D.F.
521 40 20

Ruiseñor 16
Bella Vista
A.Obregón
México, D.F.

11. Carlos A. Díaz Ramírez
Casa de Bolsa Bursamex
I. la Católica 43-801
Centro
Cuauhtémoc
06000 México, D.F.
518 54 20

Coahuila 92 D
Roma Sur
México,D.F.
564 61 02

12. Jorge Ehrlich Morin
Cía. de Luz y Fza. del Centro,S.A.
M.Ocampo 171-610
México 11310 , D.F.
566 09 21

Fdo. González Roa 41
Satélite
Naucalpan, Edo. de Méx.

13. Gustavo A. Erosa Figueroa
TELEVISA, S.A.
Av.Chapultepec 19-1° Piso
México, D.F.
588 48 14

Av.Ceracruz 113-203
Condesa
México, D.F.
286 56 07

14. Alberto Espinosa González
El Colegio de México
Camino al Ajusco 20
Pedregal de Sta. Teresa
M agdalena Contreras
01000 México, D.F.
568 60 33 Ext.192

Retorno 3 Legión del Nte. M 26 Lote 2 S
4ta.Secc. U.Ejército de Ote.
Ixtapalapa
México, D.F.

15. Jorge I EvanAvila
UNAM
Fac. de Ing.
UNAM

Carrt. México-Xoxh. 161 B-33
La Piedad
Tlalpán
México, D.F.
655 66 76

16. Víctor J. Flores Mastache
Teléfonos de México, S.A.
Parque Vía No. 190 Ofi. 950
Cuauhtémoc
México , D.F.
222 52 09

Ote. 87 No.2506
Col. E.Zapata
G.A.Madero
07850 México, D.F.
537 76 50

17. Gustavo Garza Salazar
PEMEX
Marina Nal. 329 10° Piso Torre
México, D.F.
254 38 14

Manuel Gamio 342
Iztapalapa
09460 México, D.F.
674 29 13

18. Carlos R. Gil Levy
PROCESBAC
Mérida 318
Martinica
León, Gto.
37500 México

J.B. la Salle 211
Panorama
León, Gto.
37160
70054

19. Martha E.Gómez Malagón
El Colegio de México
Camino al Ajusco 20
Pedregal de Sta. Teresa
Magdalena Contreras
01000 México, D.F.
568 60 33 Ext.396

Av. El Cántaro 11 A 102
Villa Coapa
Xochimilco
14390 México, D.F.
594 24 95

20. Francisco Góngora Flores
General Elecric de México, S.A. de C.V.
Vía Morelos 351
Estado de México
569 00 16

Alsines No. 641 Manz 112
Villa de las Flores
Coacalco, Edo. de México
915 87413355

21. J.Noe González Gómez
Casa de Bolsa Madero, S.A.
Reforma 90 2° Piso
México, D.F.
592 20 10

Vicente García Torres 75-13
Barrio la Concepción
Coyoacán
04020 México, D.F.

22. Jesús G. González Urrea
General Electric de México, S.A. de C.V.
Av. Vía Morelos 351
Cerro Gordo
Estado de México
569 00 16

Ciénega 24
Fracc. Lomas de Capistrano
Atizapán de Zaragoza
Edo. de México
398 31 66

23. Octavio Garriño Montijo
IMP
Eje Central Lázaro Cárdenas 152
G.A.Madero
567 66 00

José M. Bustillos 67 Int.201
Algarín
Cuauhtémoc
06880 México, D.F.

24. Antonio Hernández Navarro
Facultad de Ingeniería
UNAM

Puebla No. 379-5
Roma Sur
Cuauhtémoc
06760 México, D.F.

25. José L. Lemus López
SEDUE
Reforma 77-10° Piso
Tabacalera
México, D.F.
546 03 30

Plúmbago 79-1
Recreo
Atzcapotzalco 02007
México 561 96 61

26. Roberto López Fix
TELEVISA,S.A.
Arcos de Belén No. 58-4° Piso
Cuauhtémoc No 06720
México, D.F.

Lacandones Mz.2 Lote 1
Plalcoligia,Tlalpán
México, D.F.

27. Daniel Martínez Villasenor
Banco de México
Plaza Degollado 10-7° Piso
Centro
México,DF
585 13 44 Ext. 31

Pegaso 139
Prado Churubusco
Coyoacán
México,D.F.
582 08 01

28. Julio H. Milla Martínez
Comisión de Fomento Minero
Puente Tecamachalco 36
Lomas de Chapultepec
México,D.F.
540 34 00

Carbajal 7 C Depto.9
Morelos
Cuauhtémoc
06200 México,D.F.
526 16 38

29. Blanca M. Monroy Escamilla
COMERMEX
Lorenzo Boturini 203
Col. Tránsito
México, D.F.
7612011 Ext.679

Miguel Negrete105
2da. Col. del Periodista
B.Juárez
03620 México, D.F.
532 06 71

30. Adolfo Pérez Buitrón
G.Atrio de S.Fco. 53-25
Coyoacán
04320 México,D.F.
658 49 74

31. Ma. Luisa Pérez Valdespino
El Colegio de México
Camino al Ajusco 20
Pedregal de Sta. Teresa
Magadalena Cont.
01000 México,D.F.
568 60 33 Ext.396

Altamirano 72-17
San Rafael
Cuauhtémoc
06470 México,D.F.
546 92 64

32. Enrique Primo Mandujano
Coquimbo 792
Lindavista
G.A.Madero
México,D.F.
586 86 29

33. Gustavo Quintero A.

34. Miguel A. Quintero Avalos
UAM

Carpio 129-3
Sta.Ma. la Ribera
México 06400 , D.F.
547 69 35

35. Mario G. Ordaz Schroeder
ISSSTE
Subdir. de Registro y Vigencia de Derechos
Plaza de la República 6
México,D.F.
566 81 49

Playa Icacos 131 204 A
Col.Marte
Iztacalco
08880 México,D.F.
590 10 01

36. Leobardo Ramírez Flores
TELEVISA, S.A.
Av.Chapultepec 18
Doctores
06720 México,D.F.

Av.Sur 4 # 288
Oriental
México 08500 D.F.

37. Pedro Ramírez Lozano
TELEVISA, S.A.
Arcos de Belem No. 58-4° Piso
Doctores
06720 México, D.F.
7 61 78 04

Blvd. Pto. Aereo No. 465-202
Moctezuma 2da. Sección V.Carranza
México, D.F.
784 33 30

38. Raymundo H. Rangel Gutiérrez
Facultad de Ingeniería
Profesor T.C.
UNAM

Calz. Méx.-Xochimilco 161
Edif.B depto. 34
Col. la Piedad Huipulco
Tlalpán
México, D.F.

39. Agustín Raya Zarco
General Electric de Méx., S.A. de S.R.L.
Km. 17.5 Carr. Laredo
Edo. de México
55500 México
569 00 16

Uxmal 127
Narvarte
B.Juárez
03020 México, D.F.
519 21 38

56. Lucio Reyes Barranca
IMP
Av.Lázaro Cárdenas 152
G.A.Madero
07730 México, D.F.
567 66 00 Ext. 20414

Amatista 35
Estrella
G.A.Madero
07810 México,D.F.
759 20 45

57. José Reyes Andrade
PEMEX
Marina Nal. 329
México, D.F.
254 38 14

58. Gerardo Reyes Retana Dahl
Secretaría de Desarrollo Urbano
Constituyentes 947
Flores de Belén
A.Obregón
México, D.F.
516 31 88

Camino Real a Tetepan 137 B
A.Obregón
01700 México, D.F.
595 84 96

59. Guillermo Ríos Coman
Misión de Sto. Domingo 9
Fracc. Misiones
Naucalpan, Edo. de Méx.
53140
562 03 12

60. Clemente Rodríguez Barrón
Universidad Nacional Autónoma de México
UNAM
México, D.F.
550 52 15 Ext.3404

Constitución de 1917
José Rdz. Glz. 9
Iztapalapa
06960 México, D.F.
691 08 48 691 08 48

61. Sergio J. Rodríguez Bejarano
Av.Marina Nal. 329
PEMEX
México, D.F.
250 26 11 Ext.23566

Jardín 3 Edif. D No.103
Naucalpan de Juárez
Edo. de Méx.
358 40 87

61. Mario Rodríguez Garduño
SEDUE
Reforma 77
México, D.F.
591 18 69

Calle Kimbila M. 110 Lote 5
Col. Lomas de Padierna
Tlalpán
14240 México, D.F.

62. José M. Saldaña Lobera
DECFI
Tacuba 5
México, D.F.
521 40 20

Sassoferrato 57
Alfonso XIII
06014 México, D.F.
563 57 82

63. Enrique Sánchez Estrada
PEMEX
Marina Nal. 329
México, D.F.
254 20 44 Ext. 5018

Viveros de Atizapán 76
Viveros de la Loma
Tlalnepantla, Edo. de Méx.
397 04 59

64. José R. Valdivieso Rosado
Cía. de Luz y Fza. del Centro, S.A.
M. Ocampo 171-610
México, D.F.
566 09 21

Dr. Atl 255-32
Sta. Ma. la Ribera
México, D.F.

65. Salomón Vargas Soto
Inst.de Relaciones Culturales, S.C..
Gabriel Mancera 1416
Valle
México, D.F.
03100 C.P.
575 52 01

C.Ote. 217 A # 19
Agrícola Oriental
Iztacalco
08500 México, D.F.

66. Silvino Vázquez Arrollo
TELMEX, S.A.