

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

"DISEÑO, E IMPLEMENTACION DEL SISTEMA DE AUTOMATIZACION DE LA FONOTECA ALEJANDRO GOMEZ ARIAS DE RADIO UNAM"

T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACION

PRESENTAN:

HECTOR R. BALCAZAR PONCE
OSCAR RUELAS VELA



DIRECTOR DE TESIS:

ING. AURELIO SANCHEZ VACA

CIUDAD UNIVERSITARIA, MEXICO, D. F. **2005**

ÍNDICE

INTRODUCCIÓN

Objetivos 4
Perspectiva general de la Tesis 5

CAPÍTULO I ANTECEDENTES

1.1 Organización de Radio UNAM 10

1.2 Fonoteca Alejandro Gómez Arias 12
1.2.1 Historia de la fonoteca 12
1.2.1 Funciones e importancia de la fonoteca 14
1.2.3 Problemática y necesidades de la fonoteca 14

CAPÍTULO II MARCO TEÓRICO

2.1 Procesos de desarrollo de un sistema de información 20
2.1.1 Modelo del proceso de desarrollo de un sistema de información 21
2.1.1.1 Ciclo de vida en cascada 21
2.1.1.2 Ciclo de vida de construcción de prototipos 26
2.1.1.3 Ciclo de vida en espiral 27
2.1.1.4 Ciclo de vida clásico o modificado 29

2.2 Modelo de datos 29

2.3 Modelo relacional 33
2.3.1 Estructura de datos 34
2.3.2 Integridad de datos 37
2.3.3 Manejo de datos 39
2.3.3.1 Álgebra relacional 39
2.3.3.2 Cálculo relacional 41

2.4 Bases relacionales 43
2.5 Redes y comunicación 57
2.6 Sistemas operativos 70
2.7 Lenguajes de programación 73

CAPÍTULO III PROPUESTA DE SOLUCIÓN

3.1 Requerimientos de fonoteca 83
3.2 Requerimientos del sistema 90
3.3 Infraestructura computacional 91
3.4 Opciones de solución y elección optima 92

ÍNDICE

CAPÍTULO IV DISEÑO, DESARROLLO E IMPLEMENTACIÓN DEL SISTEMA

4.1 Aplicación de la metodología elegida 98
4.1.1 Diagramas de contexto 98
4.1.2 Diagrama de flujo de datos 99
4.1.3 Diccionario de datos 103
4.1.4 Modelo entidad-relación 111
4.1.5 Normalización 115

4.2 Diseño y construcción del Back End 118
4.3 Diseño y construcción del Front End 125
4.4 Pruebas de integración del sistema 139
4.5 Factibilidad operativa 147
4.6 SQL Server 153

CAPÍTULO V CONCLUSIONES

5.1 Conclusiones 163

APÉNDICES

A. GLOSARIO
B. MANUAL DE USUARIO
C. ERWIN

BIBLIOGRAFÍA

INTRODUCCIÓN

INTRODUCCIÓN

INTRODUCCIÓN

Las técnicas de grabación dieron al mundo sonoro, lo que la escritura a la palabra oral: la posibilidad de permanencia y de reproducción múltiple y simultánea. Facilitan, sin duda, el registro de composiciones sonoras de la mayor sencillez pero también con un rango amplio de complejidad y matización, lo cual supone el dominio eficaz de las tecnologías.

La grabación no sólo reproduce el mundo sonoro que le rodea. También da voz a la palabra y restituye la oralidad de los enunciados elaborados y grabados por los hablantes. Esto es marcadamente fértil en el canto. La imprenta y la grabadora intensifican la oralidad primaria en cuanto “a su mística de la participación, su insistencia en un sentido comunitario, su concentración en el momento presente, e incluso su empleo de fórmulas”. Si bien los fonoregistros obedecen a una particularización de los contextos próximos, facilitan a su vez la vinculación con otras culturas y sus mundos sonoros. Tienden puentes relacionantes que casi no conocen límites salvo el acceso al equipo necesario.

La acumulación de fono registros de diversa índole, y la necesidad de su sistematización para manejarlos en beneficio del público interesado, y de estudiantes e investigadores, dio pie a la creación de las fonotecas, en estrecha relación con los archivos sonoros preexistentes.

El concepto de fonoteca corresponde a todo el material inédito, grabado en campo (cintas de carrete abierto, cassettes, dats, minidiscs, fotografías, videos), y a lo grabado, ya editado y publicado -que puede o no ser de campo- (discos de acetato, discos compactos, cassettes, vídeos), como definición se tiene: colección o archivo de cintas o alambres magnetofónicos, discos, etc., impresionados con la palabra hablada, con música u otros sonidos. Real Academia Española

INTRODUCCIÓN

Como tales, las fonotecas tienen una historia relativamente reciente, aparecen a fines del siglo XIX y comienzos del siglo XX y surgen desde los inicios con diversas orientaciones, Díaz Viana afirma que “El interés por las músicas exóticas y por coleccionar los singulares instrumentos ligados a ellas fue en aumento a lo largo del siglo XIX, pero también se produjo, sobre todo en la segunda mitad de esa centuria, una importante corriente “folklórico-romántica” que favorecía las recopilaciones de música –fundamentalmente cantos— transmitida por los campesinos europeos”.

De acuerdo con una encuesta nacional llevada a cabo por Radio Educación para conocer el estado de las fonotecas en México, se comprobó que en la mayoría de las estaciones no existen condiciones adecuadas para la catalogación y preservación de los documentos sonoros. La situación más grave se detectó en las radiodifusoras comerciales, pero a sus propietarios no parece importarles mucho esta situación.

La insuficiencia de recursos económicos, la escasa preparación del personal y la ausencia de voluntad política son aspectos que inciden en la actual situación de las fonotecas de México que, en palabras del investigador Fernando Osorio, requieren de una atención urgente y prioritaria.

Objetivo

Analizar, diseñar y desarrollar un sistema computacional de manejo de base de datos, que proporcione una solución adecuada a la problemática y necesidades de la fonoteca “Alejandro Gómez Arias” de Radio UNAM.

Perspectiva General de la Tesis

Capítulo I, Antecedentes, se presenta un panorama general de Radio UNAM y la Fonoteca Alejandro Gómez Arias en la cual se implementará el sistema a desarrollar, considerando sus objetivos y su problemática.

Capítulo II, Marco Teórico, se presentan los conceptos para el desarrollo de un sistema basado en computadora de manejo de base de datos, así como los conceptos de IT con los que se relaciona el sistema a desarrollar.

Capítulo III, Propuesta de Solución, en este Capítulo se presentan los requerimientos de la Fonoteca, los recursos con que cuenta Radio UNAM y la elección de ellos para el desarrollo del sistema.

Capítulo IV, Diseño, desarrollo e implementación del sistema, se presenta la solución a los requerimientos de la Fonoteca mediante el análisis, diseño y construcción del software, evaluando el sistema mediante pruebas de integridad.

Capítulo V, Conclusiones, se muestran las conclusiones a las que se llegaron después de la realización de este trabajo.

En los apéndices se incluyen el glosario de términos y el manual de usuario.

ANTECEDENTES

CAPÍTULO I

1. ANTECEDENTES

La información es un insumo esencial para las actividades en la vida universitaria, de ahí la necesidad de fortalecer los mecanismos de cuidado y manejo del acervo nacional, con el que cuenta la Fonoteca “Alejandro Gómez Arias” de Radio UNAM; pues constituye parte fundamental no solo en la vida universitaria sino del patrimonio nacional.

Radio UNAM ha permanecido abierta a todas las corrientes del pensamiento y a la crítica ejercida por hombres de todas las ideologías. Se preservan los testimonios de connotados humanistas como Miguel León Portilla, León Felipe, Luis Rius, Juan García Ponce, Miguel Ángel Asturias y Juan Rulfo, sólo por mencionar algunos ejemplos.

En los últimos años el uso de la tecnología se ha convertido en una forma de solventar el trabajo exhaustivo de las personas, es así como se han generado sistemas computacionales que permiten realizar las labores cotidianas con una mayor eficiencia.

Radio UNAM inició sus transmisiones en 1937 con las siglas XEXX, en el cuadrante 1170 (Khz) de Onda Media, ahora Amplitud Modulada. Su potencia era de 5 mil watts y la programación constituía cuatro horas diarias de transmisión.

Actualmente transmite en el cuadrante XEUN 96.1MHz de la frecuencia modulada y el 860KHz de la amplitud modulada, además de XEYU 9600KHz en onda corta y esta incursionando con sus primeras transmisiones por Internet.

Es la radiodifusora cultural de mayor tradición en el país. Como radio universitaria y cultural, la programación pretendió desde sus inicios poner al alcance del pueblo, los logros culturales nacionales y los recursos de la cultura de todos los tiempos y todos los países, así como:

- Servir de vínculo y enlace entre los universitarios y con la sociedad en general.
- Participar en las tareas de extensión de la cultura y el quehacer universitario.

- Investigar nuevos métodos de producción radiofónica y, a través de éstos, experimentar formas diversas de expresión y difusión didáctica y cultural.

1.1 Organización de Radio UNAM

Radio Universidad con el fin de ofrecer sus servicios de manera eficiente tiene una organización estructurada la cual delega tareas definidas a cada una de sus dependencias las cuales se encargan del objetivo común de la difusión de la cultura, esta estructura se muestra en el siguiente organigrama.

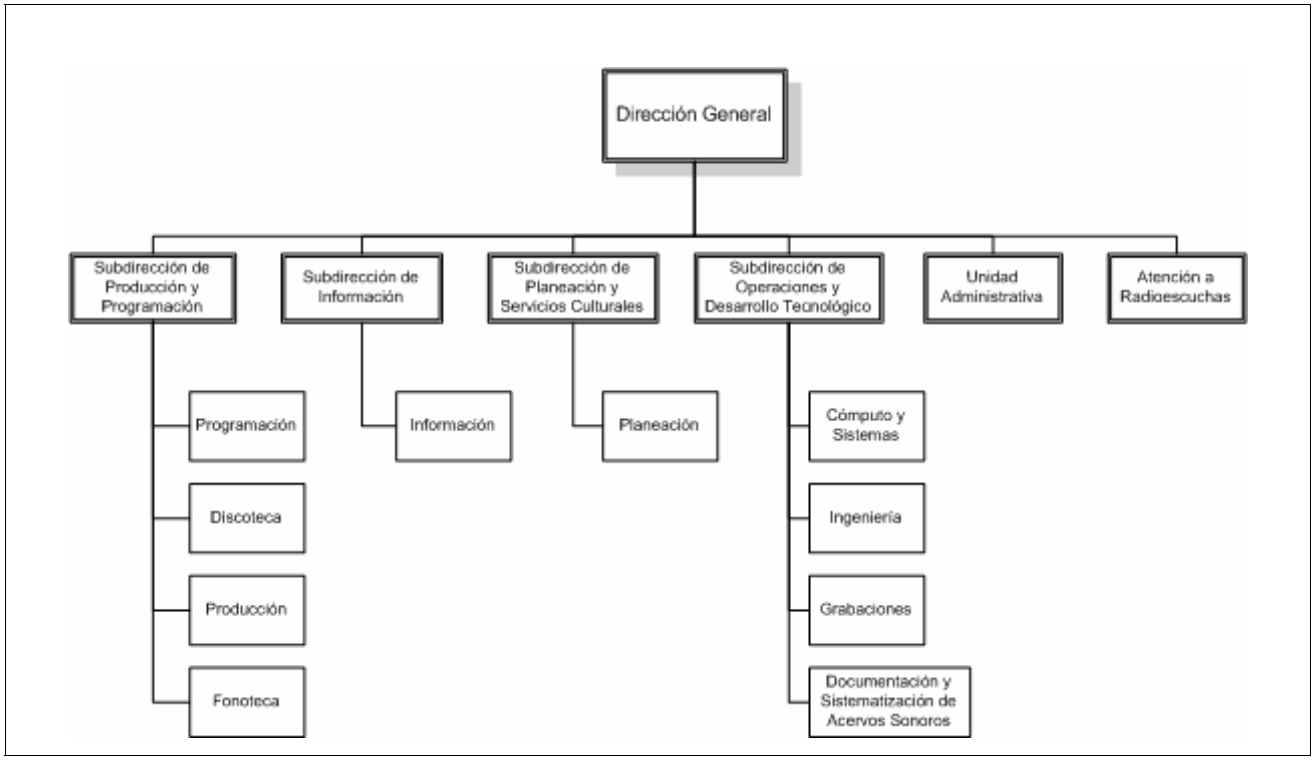


Figura 1.2.1 Organigrama Radio UNAM

Dirección General; Mtro. Fernando Álvarez del Castillo A.

Subdirección de Producción y Programación; Lic. María Teresa Moya Malfavón

Jefe de Programación; Lic. Nashrú López Rascón

Jefe de Discoteca; Lic. Eduardo González de Salceda U.

Jefe de Producción; Lic. María de la Paz Gener Rivera

Jefe de Fonoteca; Yolanda Medina Delgado

Subdirección de Información; Lic. Antonio Morales Cortés

Jefe de Información; Lic. Cutberto Domínguez Rodríguez

Subdirección de Planeación y Servicios Culturales; Prof. Carlos Narro Robles

Jefe de Planeación; Mtra. Josefina King

Subdirección de Operaciones y Desarrollo Tecnológico; Ing. Armando Trujillo Pantoja

Jefe de Cómputo y Sistemas; Ing. Oscar de Jerónimo García

Jefe de Ingeniería; Ing. Eusebio Mejia Yerves

Jefe de Grabación; Ing. Manuel Garro García

Jefe de Documentación y Sistematización de Acervos Sonoros; Lic. Victor H. Ruíz V.

Unidad Administrativa; Lic. Rodolfo Consuegra Reyes

Localización: Radio UNAM se localiza en la calle de Adolfo Prieto No. 133 en la Colonia del Valle.



Figura 1.2.2 Localización Radio UNAM

1.2 Fonoteca Alejandro Gómez Arias

1.2.1 Historia de la fonoteca

En sus inicios Radio UNAM (1937), solo realizaba programas en vivo, ya que no contaba con la tecnología necesaria para la grabación y edición de audios, por lo que no existe material de estos primeros años de existencia.

Cuando Radio UNAM se localiza en Justo Sierra se realizan trabajos de producción de series (año 1954), con lo que se obtuvo el primer material grabado.

En el año 1958, cuando las instalaciones se trasladan a Ciudad Universitaria, la Facultad de Arquitectura proporciona una cabina de grabaciones en las mañanas, y por las tardes (13 horas) comenzaban las trasmisiones de los programas previamente grabados.

Los programas grabados eran seleccionados por Rodolfo Sánchez Alvarado y los considerados

relevantes y los que pudieran tener una utilidad en el futuro se guardaban ocultándolos de la dirección, pues las cintas se reciclaban, perdiéndose así gran cantidad de programas, de Juan José Arreola, Carlos Fuentes (rescatando de este último solo una intervención) y la “Antología Caprichosa“ de Octavio Paz, de 1957, la cinta más antigua encontrada.

Al llegar Max Aub a la dirección de Radio Universidad se empezó a realizar mucha producción grabada, por lo que La Facultad de Arquitectura les sede la cabina a la emisora, ampliando con esto los horarios de transmisión, surge entonces el departamento de grabación.

A partir de la década de los setentas nace la conciencia de tener un archivo en donde se puedan localizar programas referentes a cuestiones de interés para la realización de otros programas.

Durante los actos conmemorativos del Quincuagésimo Aniversario de la fundación de Radio UNAM se inauguró la Fonoteca “Alejandro Gómez Arias”, como reconocimiento al líder de la Autonomía Universitaria y primer Director de la emisora.

En la década de los ochentas se le da un verdadero interés a la fonoteca, organizando todas las cintas para facilitar la búsqueda de los materiales necesarios, actualmente por medio de series y se van anexando los nuevos programas, registrándolos con un número de serie y otro correspondiente al programa.

La fonoteca cuenta con un estimado de 120 mil soportes, presentados en diferentes formatos, como son cintas de carrete abierto, discos compactos, DATs, casetes y muy pocos minidiscos, con un crecimiento de aproximadamente 3 mil soportes por año.

El auge de la ciencia y la técnica en nuestro siglo ha abierto la posibilidad de salvar del olvido las realizaciones acústicas, testimonio de esto son los miles y miles de soportes de la Fonoteca “Alejandro Gómez Arias”, sin duda alguna, una de las más valiosas de nuestro país.

El mantenimiento del material magnetofónico, el registro computarizado y su catalogación constituyen sustanciales avances para la investigación, el intercambio y enriquecimiento de la programación radiofónica de estaciones hermanas.

1.2.2 Funciones e importancia de la fonoteca

La función principal de este recinto es el de mantener resguardadas, ordenadas y bien distribuidas, las más de treinta y dos mil cintas en las que se encuentran el testimonio cultural de la Universidad y de algunos episodios importantes que forman parte de la historia del México contemporáneo.

Además de realizar el manejo rápido y adecuado de toda la información que se tiene, clasificándola de acuerdo a la política del departamento de fonoteca.

Cuenta actualmente con un acervo de alrededor de 120 mil soportes (grabaciones, en distintos formatos) todas ellas clasificadas en fichas creadas por el personal de fonoteca (ficha fonotográfica), teniendo un crecimiento de 3000 a 3500 soportes por año, colocados en una bóveda especial para su cuidado.

Proporciona los soportes requeridos para diversas actividades, como son la investigación y la difusión principalmente.

Administra los soportes para su acomodo y su manejo dentro y fuera de sus instalaciones.

Para la consulta de la información de los soportes cuenta con un sistema computacional desarrollado en 1998 por la Universidad de Colima.

1.2.3 Problemática y necesidades de la fonoteca

La Fonoteca enfrenta el problema de manejar la información de los soportes de manera rápida y precisa, debido a que la herramienta con la que cuenta proporcionada por la Universidad de Colima, es ineficiente, ya que no cumple con los siguientes puntos:

- Reducir el tiempo de búsqueda de los soportes requeridos, ubicándolos de manera física.
- Manejar la gran cantidad de información que se tiene y que esta en crecimiento constante, de manera precisa y clara.
- La seguridad de la información, haciéndola confiable, de manera que únicamente personal autorizado puede actualizar los datos e impedir alteraciones a los datos por parte usuarios que consulten a la herramienta computacional.
- Interacción amigable con el usuario
- Administración total por parte del personal del área de sistemas en las instalaciones de Radio UNAM.

La Fonoteca de Radio UNAM, tiene la necesidad de obtener un sistema basado en herramientas computacionales, que logre cumplir con los puntos anteriores. De esta manera surge el proyecto SFAGA (Sistema Fonoteca Alejandro Gómez Arias), un sistema basado en computadora que utiliza una base de datos, cuyo objetivo es el de resolver la problemática a la que se enfrenta la Fonoteca de Radio UNAM.

En los siguientes capítulos, se dan los fundamentos y requerimientos, para el desarrollo de sistemas computacionales que utilizan una base de datos relacional, que permitirán el desarrollo del proyecto mencionado.

MARCO TEÓRICO

CAPÍTULO II

2 Marco Teórico

Un sistema computacional, permite automatizar tareas que el ser humano realiza, haciendo que estas tareas se realicen con mayor rapidez y exactitud, evitando en la menor proporción posible, errores que pueden desencadenar en pérdidas importantes de información, estos sistemas computacionales se apoyan de herramientas de almacenamiento de información conocidas como bases de datos, que permiten organizar la información y hacerla disponible, de manera que junto con una interfaz de usuario, dicho usuario pueda visualizar de manera lógica la información almacenada en la base de datos.

SFAGA será un sistema basado en computadora de manejo de base de datos relacional, lo cual permitirá resolver la problemática que se presenta en la Fonoteca de Radio UNAM. Este capítulo fundamenta los conceptos para el desarrollo de SFAGA.

El proceso denominado ingeniería de sistemas basados en computadora, se concentra en una variedad de elementos, analizando, diseñando y organizándolos de manera tal que puedan ofrecer un producto, un servicio o una tecnología para la manipulación de la información. Los elementos anteriormente mencionados son generalmente: software, hardware, gente, bases de datos, documentación y procedimientos.

Anteriormente los sistemas se desarrollaban usando técnicas de gestión orientadas a hardware. Exigían un análisis y diseño completo antes de que algo se construyera, aplicando los controles, los métodos y las herramientas que son conocidas como ingeniería del hardware, en donde el software no era más que un agregado. Actualmente el software es el elemento principal en este tipo de sistemas. La ingeniería de software es el establecimiento y uso de principios robustos de la ingeniería a fin de obtener software económico que sea fiable y que funcione eficientemente sobre máquinas reales. La ingeniería del software pretende proporcionar un marco de trabajo para construir software de calidad.

El sistema de manejo de base de datos (DBMS, database management system), consiste en un conjunto de datos relacionados entre sí y un grupo de programas que permiten el acceso a ellos, a este conjunto de datos se le denomina base de datos y contiene información de un ente

determinado.

Los sistemas que contienen bases de datos, son diseñados para el manejo de grandes cantidades de información, lo que lleva a la definición de las estructuras para el almacenamiento y de los mecanismos para el manejo de la información.

Debido a la importancia que tiene la información en las organizaciones, las bases de datos son un recurso valioso, lo cual ha conducido al desarrollo de un gran número de conceptos y técnicas para manejar datos de forma eficiente. Actualmente, el software ha superado al hardware como la clave del éxito de muchos sistemas basados en computadora. Lo que diferencia a una compañía de su competidora es la eficiencia y oportunidad de la información dada por el software y las bases de datos relacionadas.

2.1 Procesos de Desarrollo de un Sistema de Información

El sistema de información es un conjunto de elementos que están organizados para realizar un objetivo predefinido procesando información, este conjunto de elementos debe incorporar una metodología de desarrollo (modelo del proceso ó paradigma), dicha metodología se selecciona de acuerdo a la naturaleza del objetivo del sistema de información el cual tiene como elementos:

SOFTWARE: Los programas de computadora, las estructuras de datos y la documentación asociada, que sirven para realizar el método lógico, procedimiento o control requerido.

HARDWARE: Los dispositivos electrónicos (ejemplo: CPU, memoria) que proporcionan la capacidad de computación y los dispositivos electromecánicos (ejemplo: sensores, motores, bombas) que proporcionan las funciones del mundo exterior.

GENTE: Los individuos que son usuarios y operadores del software y del hardware.

BASES DE DATOS: Una colección grande y organizada de información a la que se accede mediante el software y que es una parte integral del funcionamiento del sistema.

DOCUMENTACIÓN: Los manuales, los impresos y otra información descriptiva que explica el uso y/o la operación del sistema.

PROCEDIMIENTOS: Los pasos que definen el uso específico de cada elemento del sistema.

Independientemente de la metodología que se utilice, del área de aplicación, del tamaño y la complejidad del sistema, el proceso de desarrollo del sistema basado en computadora lleva consigo una serie de fases genéricas que son definidas a continuación:

Definición: Durante esta fase, el que desarrolla el sistema intenta identificar qué información ha de ser procesada, qué función y rendimiento se desea, qué interfaces han de establecerse, qué restricciones de diseño existen y qué criterios de validación se necesitan para definir un sistema correcto.

Desarrollo: Quién desarrolla el software intenta descubrir cómo han de diseñarse las estructuras de datos, cómo ha de traducirse el diseño a un lenguaje de programación y cómo ha de realizarse la prueba.

Mantenimiento: Se centra en el cambio que va asociado a la corrección de errores, a las adaptaciones requeridas por la evolución del entorno del sistema y a las modificaciones debidas a los cambios de los requisitos del cliente, dirigidos a reforzar o ampliar el sistema.

2.1.1 Modelo del Proceso de Desarrollo de un Sistema de Información

Los modelos del proceso para el desarrollo de un sistema tienen relación con el ciclo de vida de este, por ciclo de vida, se entiende la sucesión de etapas por las que pasa el sistema de información desde que es concebido hasta que se deja de usar. Dichas etapas llevan asociadas una serie de tareas que deben realizarse, y una serie de documentos que serán la salida de

cada etapa y servirán a su vez de entrada para la etapa siguiente. El propósito del ciclo de vida es: planear, ejecutar y controlar el desarrollo de un sistema. El ciclo de vida define las etapas y las tareas esenciales para el desarrollo del sistema, sin importar el alcance del sistema que se intenta desarrollar.

Existen diversos ciclos de vida y cada uno de ellos tiene asociado una serie de métodos, herramientas y procedimientos, los más utilizados son:

1. Ciclo de vida clásico o modelo en cascada
2. Construcción de prototipos
3. Modelo en espiral o evolutivo
4. Ciclo de vida clásico modificado

Los cuales son descritos a continuación.

2.1.1.1 Ciclo de Vida en Cascada

Este modelo es el más antiguo de los empleados en la Ingeniería de Sistemas y se desarrolló a partir del ciclo convencional de una ingeniería, exige un enfoque sistemático y secuencial del desarrollo del sistema de información, que comienza en el nivel de la ingeniería de sistemas y avanza a través de etapas secuénciales sucesivas. Estas etapas son las siguientes:

Ingeniería y análisis del sistema.

El software es siempre parte de un sistema mayor, por lo que siempre va a interrelacionarse con otros elementos, ya sea hardware, máquinas o personas. Por esto, el primer paso del ciclo de vida del sistema de información consiste en un análisis de las características y el comportamiento del sistema del cual el software va a formar parte. En el caso del desarrollo de un nuevo sistema, se deben analizar cuáles son los requisitos y la función del sistema, después

se asigna un subconjunto de estos requisitos al software. En el caso de un sistema ya existente se analiza el funcionamiento de este, las operaciones que se llevan a cabo, y se asigna al software aquellas funciones que se van a automatizar.

La ingeniería del sistema comprende, por tanto, los requisitos globales a nivel del sistema, así como una cierta cantidad de análisis y de diseño a nivel superior, es decir sin entrar en mucho detalle.

Análisis de requisitos del software.

El análisis de requisitos debe ser más detallado para aquellos componentes del sistema que se va a implementar mediante software. El ingeniero del software debe comprender cuáles son los datos que se van a manejar, cuál va a ser la función que tiene que cumplir el software, cuáles son los interfaces requeridas y cuál es el rendimiento que se espera lograr.

Los requisitos, tanto del sistema como del software deben documentarse y revisarse con el cliente.

Diseño.

El diseño es el proceso que traduce los requisitos en una representación del software de forma que pueda conocerse la arquitectura, funcionalidad e incluso la calidad del mismo antes de comenzar la codificación.

El diseño se aplica a cuatro características distintas del software: la estructura de los datos, la arquitectura de las aplicaciones, la estructura interna de los programas y las interfaces.

Al igual que el análisis de requisitos del software, el diseño debe documentarse y formar parte de la configuración del software, el control de configuraciones es lo que permite realizar cambios en el software de forma controlada.

CAPÍTULO II

Codificación.

La codificación consiste en la traducción del diseño a un formato que sea legible para la máquina. Si el diseño es lo suficientemente detallado, la codificación es relativamente sencilla, y puede hacerse de forma automática, usando generadores de código.

Prueba.

Una vez que ya se tiene el programa ejecutable, comienza la fase de pruebas. El objetivo es comprobar que no se hayan producido errores en alguna de las fases anteriores, especialmente en la codificación. Para ello deben probarse todas las sentencias y todos los módulos que forman parte del sistema.

Utilización.

Una vez superada la fase de pruebas, el software se entrega al cliente y comienza la vida útil del mismo. La fase de utilización conlleva a fases posteriores (mantenimiento y sustitución) y dura hasta que el software, ya reemplazado por otro, deje de utilizarse.

Mantenimiento.

El software sufrirá cambios a lo largo de su vida útil. Estos cambios pueden ser debidos a tres causas:

1. Que durante la utilización, el cliente detecte errores en el software: los errores latentes.
2. Que se produzcan cambios en alguno de los componentes del sistema informático: por ejemplo cambios en la máquina, en el sistema operativo o en los periféricos.
3. Que el cliente requiera modificaciones funcionales (normalmente ampliaciones) no contempladas en el proyecto.

En cualquier caso, el mantenimiento supone volver atrás en el ciclo de vida, a las etapas de codificación, diseño o análisis dependiendo de la magnitud del cambio.

Sustitución.

La vida del software no es ilimitada y cualquier aplicación, por buena que sea, acaba por ser sustituida por otra más amplia, más rápida o fácil de usar.

La sustitución de un software que está funcionando por otro que acaba de ser desarrollado es una tarea que hay que planificar cuidadosamente y que hay que llevar a cabo de forma organizada. Es conveniente realizarla por fases, si esto es posible, no sustituyendo todas las aplicaciones de golpe, puesto que la sustitución conlleva normalmente un aumento de trabajo para los usuarios, que tienen que acostumbrarse a las nuevas aplicaciones, y también para los que implementan la sustitución, que tienen que corregir los errores que aparecen. Es necesario hacer una migración de la información que maneja el sistema anterior a la estructura y el formato requerido por el nuevo. Además, es conveniente mantener los dos sistemas funcionando en paralelo durante algún tiempo para comprobar que el sistema nuevo funcione correctamente y para asegurar el funcionamiento normal de la empresa aún en el caso de que el sistema nuevo falle y tenga que volver a alguna de las fases de desarrollo.

La sustitución implica el desarrollo de programas para la interconexión de ambos sistemas, el anterior y el nuevo, de manera que se permita migrar la información entre ambos, evitando duplicar del trabajo de las personas encargadas del proceso de datos, durante el tiempo en que ambos sistemas funcionen en paralelo.

Problemas de aplicación de ciclo de vida en cascada.

El ciclo de vida en cascada es el más antiguo, más conocido y más ampliamente usado en la Ingeniería de Sistemas. No obstante, ha sufrido diversas críticas, debido a los problemas que se plantean al intentar aplicarlo a determinadas situaciones. Entre estos problemas están:

El modelo en cascada, a pesar de ser lineal, contiene flujos que permiten la vuelta atrás. Así, desde el mantenimiento se vuelve al análisis, el diseño o la codificación, y también desde cualquier fase se puede volver a la anterior si se detectan fallos. Estas vueltas atrás no son controladas, ni quedan explícitas en el modelo.

En realidad los sistemas de información no siguen un ciclo de vida estrictamente secuencial como propone el modelo. Siempre hay iteraciones. El ejemplo más típico es la fase de mantenimiento, que implica siempre volver a alguna de las fases anteriores, pero también es muy frecuente que en una fase, por ejemplo el diseño, se detecten errores que obliguen a volver a la fase anterior, el análisis.

Es difícil que se puedan establecer inicialmente todos los requisitos del sistema. Normalmente los clientes no tienen conocimiento de la importancia de la fase de análisis o bien no han pensado en detalle, que es lo que requieren el software. Los requisitos se van aclarando y refinando a lo largo del desarrollo del sistema de información, según se plantean dudas concretas en el diseño o la codificación. Sin embargo, el ciclo de vida clásico requiere la definición inicial de todos los requisitos y no es fácil acomodar en él las incertidumbres que suelen existir al comienzo de todos los proyectos.

Hasta que se llega a la fase final de desarrollo: la codificación, no se dispone de una versión operativa del programa. Como la mayor parte de los errores se detectan cuando el cliente puede probar el programa, no se detectan hasta el final del proyecto. Cuando son más costosos de corregir y existen más presiones para que el programa se ponga definitivamente en marcha.

Todos estos problemas son reales, sin embargo, es mucho mejor desarrollar el software siguiendo el modelo de ciclo de vida en cascada que hacerlo sin ningún tipo de guías.

2.1.1.2 Ciclo de Vida de Construcción de Prototipos

La construcción de prototipos, es un ciclo de vida que facilita la creación del sistema de información a construir mediante una de las tres formas siguientes.

1. Un prototipo en papel o un modelo basado en computadora que describa la interacción hombre-máquina, de forma que facilite al usuario la comprensión de cómo se producirá tal interacción.

2. Un prototipo que implemente algunos subconjuntos de la función requerida del programa deseado.

3. Un programa existente que ejecute parte o toda la función deseada, pero que tenga otras características que deban ser mejoradas en el nuevo trabajo de desarrollo.

Como en todos los métodos de desarrollo de un sistema de información, la construcción de prototipos comienza con la recolección de los requisitos. Luego se produce un “diseño rápido”. El diseño rápido se enfoca sobre la representación de los aspectos del software visibles al usuario (por ejemplo, métodos de entrada y formatos de salida). El diseño rápido conduce a la construcción de un prototipo. El prototipo es evaluado por el cliente y se utiliza para refinar los requisitos del software a desarrollar. Se produce un proceso interactivo en el que el prototipo es “afinado” para que satisfaga las necesidades del cliente, al mismo tiempo que facilita al que lo desarrolla una mejor comprensión de lo que hay que hacer.

2.1.1.3 Ciclo de Vida en Espiral

El modelo en espiral ha sido desarrollado para cubrir las mejores características tanto del ciclo de vida clásico, como de la creación de prototipos, añadiendo al mismo tiempo un nuevo elemento; el análisis de riesgo. El modelo es representado mediante una espiral que define cuatro actividades principales, representadas por cuatro cuadrantes:

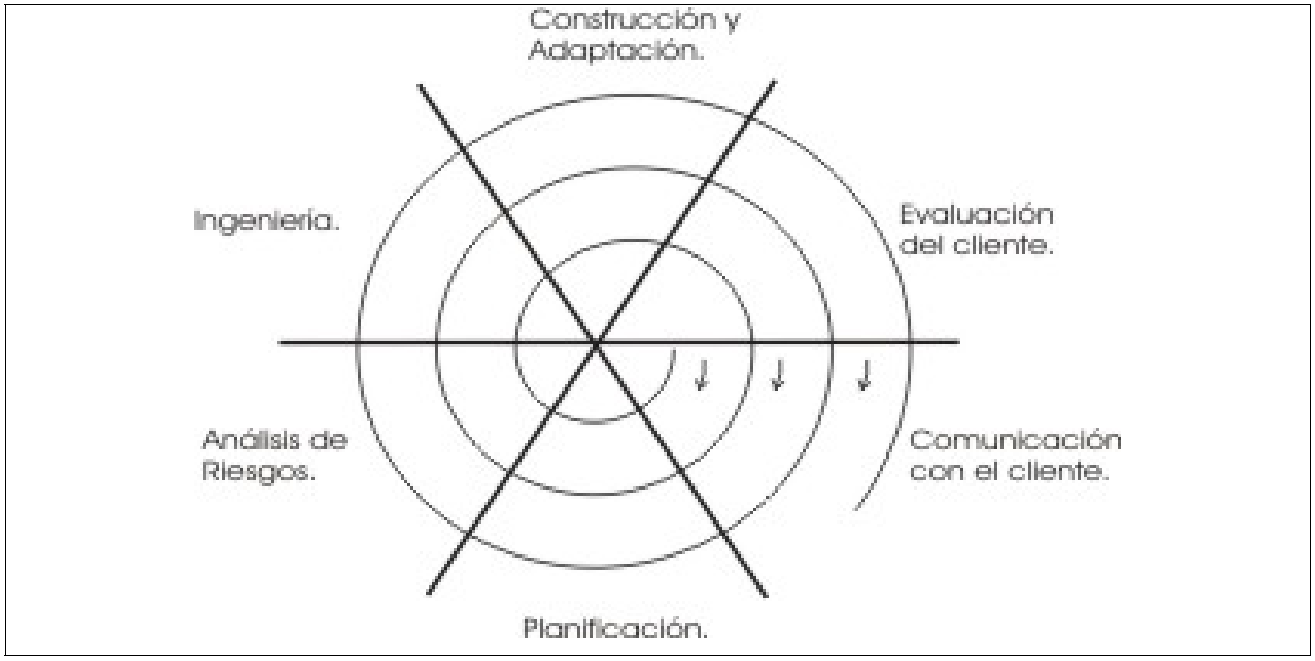


Figura 2.1.1 Ciclo de Vida en Espiral

1. Comunicación con el Cliente: Evaluación del software por parte del cliente.
2. Planificación: Determinación de objetivos, alternativas y restricciones.
3. Análisis de riesgo: Análisis de alternativas e identificación y/o resolución de riesgos.
4. Ingeniería: Desarrollo del producto de "siguiente nivel".
5. Construcción y adaptación: Tareas requeridas para construir, probar, instalar y proporcionar soporte al usuario (Documentación).
6. Evaluación del cliente: Valoración de los resultados de la ingeniería.

Con cada iteración alrededor de la espiral, se construyen sucesivas versiones del software cada vez más completas. Durante la primera vuelta alrededor de la espiral se definen los objetivos, las alternativas y las restricciones, se analizan e identifican los riesgos. Si el análisis de riesgo indica que hay una incertidumbre en los requisitos, se puede usar la creación de prototipos en el cuadrante de ingeniería para dar asistencia tanto al encargado del desarrollo como al cliente.

El cliente evalúa el trabajo de ingeniería y sugiere modificaciones. En base a los comentarios del cliente se produce la siguiente fase de planificación y de análisis de riesgo. En cada bucle alrededor de la espiral, la culminación del análisis de riesgo resulta en una decisión de “seguir o no seguir”. Si los riesgos son demasiado grandes, se puede dar por terminado el proyecto.

Cada vuelta alrededor de la espiral requiere ingeniería, que se puede llevar a cabo mediante el enfoque del ciclo de vida clásico o de la creación de prototipos.

2.1.1.4 Ciclo de Vida Clásico o Modificado

Tomando en cuenta los elementos del ciclo de vida clásico previamente expuestos, y con la intención de hacerlo aplicable a la mayoría de las situaciones, surge el ciclo de vida clásico modificado; que proporciona más flexibilidad entre las fases permitiendo iteraciones entre ellas, según sean necesarias; trata de acomodar cambios en los requerimientos y disminuir los efectos de errores en cada etapa del desarrollo.

El desarrollo de un sistema de bases de datos, requiere de una metodología a seguir, además de un modelo de datos, a continuación se describe este concepto.

2.2 Modelo de Datos

Un modelo es una representación simplificada de un sistema real, por ejemplo; una base de datos es un modelo de conjuntos de datos relativos a una organización. Un modelo de datos es una estructura general de los datos y técnicas de acceso proporcionadas por un SGBD (Sistema Gestor de Base de Datos), y se define por los componentes: Un conjunto de objetos y sus interrelaciones. Esto representa las características estáticas o invariantes e incluye las propiedades de los objetos, un conjunto de operaciones, o lenguaje que representa las

características dinámicas, restricciones sobre los objetos, sus interrelaciones y las operaciones definidas sobre ellos.

El conjunto de objetos y sus interrelaciones resultan de abstracciones realizadas sobre la estructura del sistema real y constituye de este modo la estructura del modelo.

Se han propuesto varios modelos de datos, los cuales pueden dividirse en tres grupo: Modelos lógicos basados en objetos, modelos lógicos basados en registros y modelos físicos de datos.

Los modelos lógicos basados en objetos se utilizan para describir los datos en los niveles conceptual y de visión. Se caracterizan por el hecho de que permiten una estructuración flexible y hacen posible especificar las limitaciones de los datos. Algunos de los más conocidos son:

1. El modelo entidad-relación (E-R).
2. El modelo binario.
3. El modelo semántico de datos.
4. El modelo infológico.

Los modelos lógicos basados en registros se utilizan para describir los datos en los niveles conceptual y de visión. Estos modelos sirven para especificar la estructura lógica general de la base de datos como una descripción en un nivel más alto de la implementación, los tres modelos más conocidos son:

- Modelo relacional (el más empleado). Todos los datos visibles al usuario están organizados estrictamente como tablas de valores. Todas las operaciones sobre la base de datos operan sobre esas tablas. Cada fila de una tabla es una instancia de los datos. Cada columna de una tabla es un atributo (valor indivisible que tiene significado por sí solo). Es el modelo de datos más sencillo y cercano a la forma humana de organizar la información.
- Modelo de red. Los datos se representan por medio de conjuntos de registros y las relaciones entre los datos se representan con ligas, que pueden considerarse como

apuntadores. También denominado modelo CODASYL. Fue el primero en aparecer comercialmente, a principios de los años setenta. Se caracteriza por almacenar direcciones de otros datos junto a la misma información. Es un modelo cercano al modo de almacenamiento interno de la computadora.

- Modelo jerárquico. Difiere al modelo de red en que los registros están organizados como conjuntos de árboles en vez de gráficas arbitrarias.

El modelo de datos entidad-relación es el más utilizado en la práctica, los conceptos de este modelo son de alto nivel semántico apegados a la realidad de discurso del usuario y no al del sistema. Propuesto originalmente como una herramienta conceptual útil en el diseño de una base de datos y no como modelo de datos a ser implementado en la práctica por un SGBD, este modelo es una descripción del aspecto del mundo real que el diseñador intenta capturar en un sistema de información. En este modelo, la estructura se describe por un conjunto de entidades u objetos de interés relacionados entre si por interrelaciones, es una técnica de diseño de bases de datos gráfica, que incorpora información relativa a los datos y la relación existente entre ellos, para poder así plasmar una visión del mundo real sobre un soporte informático. Sus características son:

- Reflejar tan sólo la existencia de los datos sin expresar lo que se hace con ellos.
- Es independiente de las bases de datos y de los sistemas operativos.
- Incluye todos los datos que se estudian sin tener en cuenta las aplicaciones que se van a tratar.

Las entidades se representan como rectángulos, los atributos como elipses y las relaciones como rombos.

El modelo Entidad-Relación fue propuesto por Chen a mediados de los años setenta, como medio de representación conceptual de los problemas y representar la visión de un sistema de forma global. Físicamente adopta la forma de un grafo escrito en papel al que se denomina diagrama Entidad-Relación. Sus elementos fundamentales son las entidades y las relaciones.

- Una entidad caracteriza a un tipo de objeto, real o abstracto, del problema a modelar. Toda entidad tiene existencia propia, tiene nombre y posee atributos definidos en un dominio determinado. Una entidad es todo aquello de lo que se desea almacenar información.

- Una relación es una asociación o relación matemática entre varias entidades. Se representan en el diagrama E-R mediante flechas y rombos. Cada entidad interviene en una relación con una determinada asociación (número de instancias o elementos de una entidad que pueden asociarse a un elemento de la otra entidad relacionada). Existen tres tipos posibles de asociación: uno a uno, uno a muchos y muchos a muchos, los cuales se describen a detalle en el punto 2.3 Modelo Relacional.

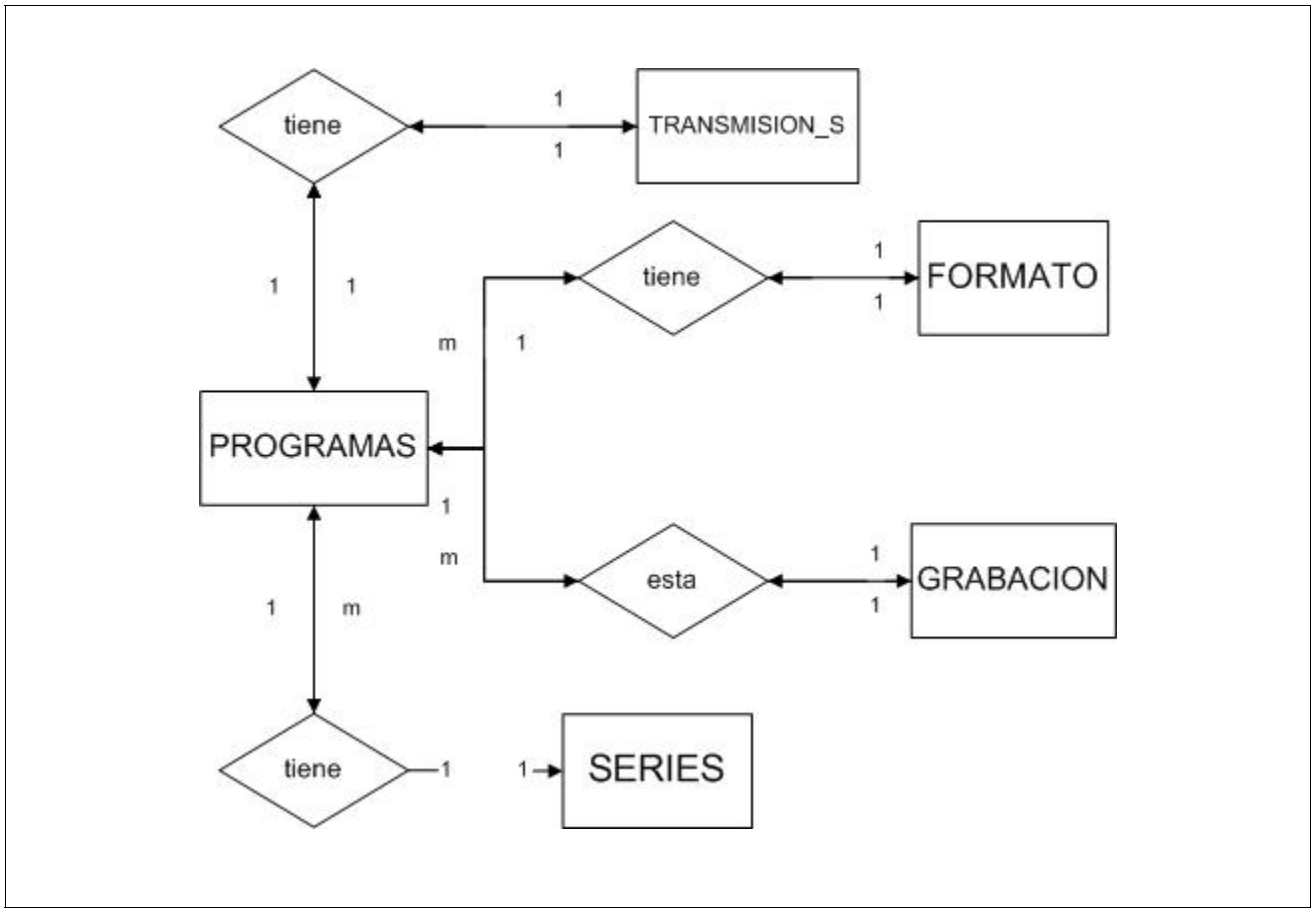


Figura 2.2.1 Diagrama Entidad-Relación

2.3 Modelo Relacional

El modelo relacional de datos es definido en 1970 por el Dr. F. Codd, por medio de una serie de reglas cuyo objetivo es lograr la independencia de la representación lógica de los datos de su almacenamiento físico.

El modelo relacional consiste en un conjunto de tablas (entidades) constituidas por filas (tuplas) y columnas (atributos), cada tabla tiene asignado un nombre único, cada atributo posee un conjunto de posibles valores llamado dominio. Una relación sobre n dominios es definida por un subconjunto del producto cartesiano de los dominios, es decir un conjunto de tuplas, las cuales se obtiene tomando un elemento de cada dominio. Una relación de n dominios se denomina rango y la cantidad de tuplas en la tabla es la cardinalidad.

El modelo relacional se define por:

1. Objetos e interrelaciones, las relaciones o tablas son el objeto básico del modelo, y las interrelaciones entre las tablas se define por los atributos comunes o compartidos.

2. Conjunto de operaciones, las más comunes son el de selección, proyección, unión, diferencia y el producto cartesiano.

3. Restricciones, algunas de las restricciones más comunes son que los valores que tomen los atributos sean atómicos, los nombres de columnas en la tabla no deben estar repetidos y las relaciones no pueden tener filas duplicadas.

El modelo relacional, como todo modelo de datos, tiene que ver con tres aspectos de los datos:

- Estructura de datos.

- Integridad de datos.

- Manejo de datos.

2.3.1 Estructura de Datos

El modelo matemático que esta bajo el modelo relacional, es la relación de teoría de conjuntos, la cual es un subconjunto del producto cartesiano o de una lista de dominios “Valores”. Está relación de teoría de conjuntos proporciona al Modelo Relacional su nombre. Los conceptos del Modelo Relacional son:

Entidad: Una entidad es un objeto concreto o abstracto que presenta interés para el sistema y sobre el que se recoge información, la cual va a ser representada en un sistema de base de datos, una entidad es la organización de los datos del objeto en forma de filas y columnas. *Una entidad es una tabla con columnas y filas*.

Las entidades tienen las propiedades:

- Tienen un nombre y éste es distinto al de las demás.
- Los valores de los atributos son atómicos: en cada tupla, cada atributo toma un solo valor. Se dice que las relaciones están *normalizadas*.
- No hay dos atributos que se llamen igual.
- El orden de los atributos no importa.
- Cada tupla es distinta de las demás.
- El orden de las tuplas no importa.

Atributo: Es una unidad básica e indivisible de información acerca de una entidad o una relación y sirve para identificar y describir a las mismas. Por ejemplo, si se va a modelar un evento como una llamada al servicio de asistencia, probablemente se querrá saber quién era el cliente, quién hizo la llamada y cuándo, así como si se resolvió o no el problema. La determinación de los atributos que hay que incluir en el modelo es un problema semántico (de significado). Se deben tomar decisiones basadas en el significado de los datos y en cómo se

utilizarán. *Un atributo es el nombre de una columna de una relación.*

Tupla: Los elementos de una relación son las tuplas o filas de la tabla. Las tuplas de una relación no siguen ningún orden. *Una tupla es una fila de una relación.*

Grado: El número de atributos de una entidad definen el grado. El grado de una relación no cambia con frecuencia. *El grado de una entidad es el número de atributos que contiene.*

Cardinalidad: En las entidades se van insertando y borrando tuplas a menudo, la cardinalidad de las mismas varía constantemente. *La cardinalidad de una relación es el número de tuplas que contiene.*

Dominio: Un dominio es el conjunto de valores que puede tomar cada uno de los atributos. La mayoría de las ocasiones se confunde dominio con tipo de datos, el ejemplo siguiente pone de manifiesto la diferencia: si se toma como atributo para una entidad *persona* su grupo sanguíneo, está claro que el tipo de dato que se tiene que usar será de tipo cadena, sin embargo, los únicos valores que puede tomar son los del conjunto {A+, A-, B+, B-, AB+, AB-, 0+, 0-}; este sería el dominio del atributo grupo sanguíneo. *Un dominio es el conjunto de valores de uno o varios atributos.*

Ya que en una tabla no hay tuplas repetidas, éstas se pueden distinguir unas de otras, es decir, se pueden identificar de modo único. La forma de identificarlas es mediante las llaves candidatas, compuestas, primarias y foráneas.

Llave candidata: Atributo o atributos que pueden distinguir de forma unívoca una tupla dentro de una tabla. Puede haber varias llaves candidatas para distinguir una misma entidad. Se elegirá como clave candidata aquel atributo que posea un dominio en el que se tenga valores únicos. Si esto no es posible, entonces se usa como llave candidata la combinación de varios atributos, de manera que esta combinación sí sea única, lo que se conoce como **Llave compuesta** o **clave compuesta**.

Clave primaria o Llave primaria: Aquella de las llaves candidatas que es designada para distinguir de forma unívoca una tupla dentro de una tabla.

Clave foránea o Llave foránea: Se trata de un atributo que es llave primaria en otra tabla.

Las anteriores llaves tienen su importancia en el modelo de la base de datos, cumpliendo con las reglas de integridad.

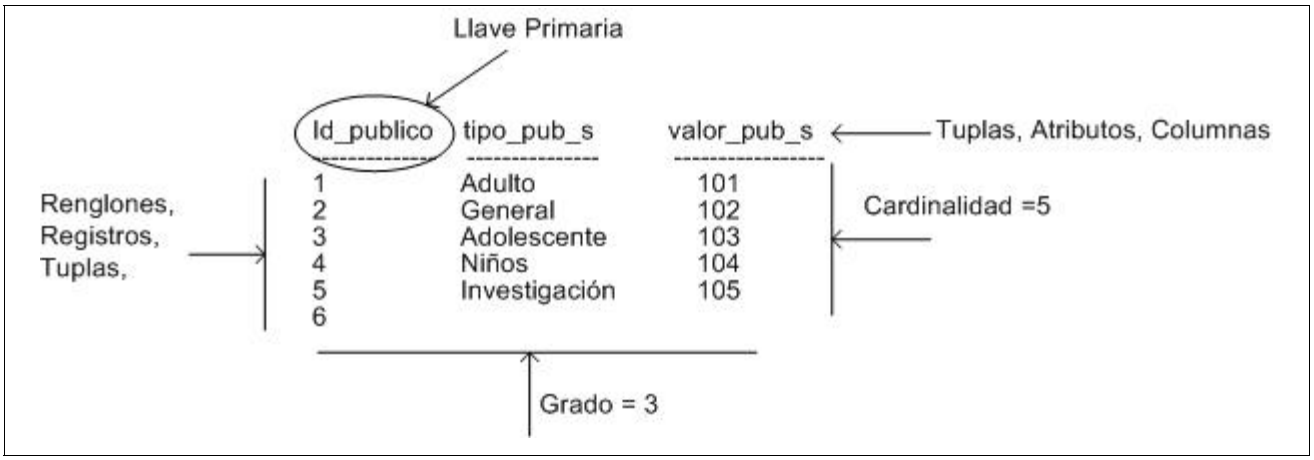


Figura 2.3.1 Conceptos del Modelo Relacional

Asociaciones entre entidades

Además de los atributos de cada entidad, un modelo de datos debe especificar las asociaciones existentes entre las entidades. Estas asociaciones son las relaciones entre entidades. Las asociaciones entre dos entidades cualesquiera pueden ser de tres tipos: uno-a-uno, uno-a-muchos y muchos-a-muchos.

Asociaciones uno-a-uno: En este tipo de relación, una vez fijado un elemento de una entidad se conoce la otra, entonces se decide que la asociación es uno-a-uno. Dado un registro de una entidad A está asociada únicamente con un registro de la entidad B y un registro de la entidad B está asociada sólo con un único registro de A.

Asociaciones uno-a-muchos**:** Es el tipo de asociación más común, donde un solo registro de una entidad se puede asociar con cero, uno o muchos registros de otra entidad. Por ejemplo, un padre puede tener varios hijos, y un hijo únicamente tiene un padre. Un registro de la entidad A está relacionado con cualquier número de registros de la entidad B, pero un registro de la

entidad B puede asociarse únicamente con un registro de la entidad A.

Asociaciones muchos-a-muchos: Las personas tienen muchas tarjetas de crédito, una tarjeta de crédito la tienen muchas personas. Este tipo de asociación no se puede modelar directamente en una base de datos relacional, se modela usando una tabla intermedia que tenga una asociación uno-a-muchos con cada uno de los participantes originales.

Tabla relacional: Es una tabla que debe cumplir las siguientes características:

- Cada fila debe ser única.
- Cada columna debe ser única.
- Los valores de las columnas deben pertenecer al dominio de cada atributo.
- Debe tener un solo tipo de fila, cuyo formato está definido por el esquema de la tabla o relación.
- El valor de la columna para cada fila debe ser único.

En los siguientes capítulos se verán conceptos fundamentales sobre el diseño de bases de datos: operaciones sobre tablas y normalización. Estos conceptos aportarán una serie de técnicas que ayudarán a que el diseño de la base de datos sea el más óptimo posible.

2.3.2 Integridad de Datos

Una vez definida la estructura de datos del modelo relacional, es necesario conocer las reglas de integridad que los datos almacenados en dicha estructura deben cumplir para garantizar que son correctos.

Al definir cada atributo sobre un dominio se impone una restricción sobre el conjunto de valores permitidos para cada atributo. A este tipo de restricciones se les denomina *restricciones de*

dominios. Hay además dos reglas de integridad muy importantes que son restricciones que se deben cumplir en todas las bases de datos relacionales y en todos sus estados o instancias (las cuales se deben cumplir todo el tiempo). Estas reglas son la *regla de integridad de entidades* y la *regla de integridad referencial*. Para poder definir estas reglas de integridad es necesario mencionar el concepto de valor nulo.

Se dice que se tiene un valor nulo cuando en una tupla un atributo es desconocido. Un nulo no representa el valor cero ni la cadena vacía, éstos son valores que tienen significado. El nulo implica ausencia de información, bien porque al insertar la tupla se desconocía el valor del atributo, o bien porque para dicha tupla el atributo no tiene sentido. Ya que los nulos no son valores, deben tratarse de modo diferente, lo que causa problemas de implementación.

Regla de integridad de entidades

La primera regla de integridad se aplica a las llaves primarias de las tuplas y expresa:

Ninguno de los atributos que componen la clave primaria puede ser nulo.

Por definición, una llave primaria es un identificador irreducible que se utiliza para identificar de modo único las tuplas. La palabra irreducible significa que ningún subconjunto de la llave primaria sirve para identificar las tuplas de modo único. Si se permite que parte de la llave primaria sea nula, se está diciendo que no todos los atributos son necesarios para distinguir las tuplas, con lo que se contradice la irreducibilidad. Esta regla sólo se aplica a las llaves primarias.

Regla de integridad referencial

La segunda regla de integridad se aplica a llaves foráneas, y expresa: *si en una relación hay alguna llave foránea, sus valores deben coincidir con valores de la llave primaria a la que hace referencia, o bien, deben ser completamente nulos*.

La llave foránea se puede definir como un atributo de A de una relación R1 y cada valor de A en R1 debe ser igual a V, donde V es el valor de la llave primaria de alguna relación R2.

2.3.3 Manejo de Datos

El tercer aspecto de un modelo de datos es la manipulación de estos. Son varios los lenguajes utilizados por los SGBD relacionales para manejar las relaciones. Algunos de ellos son *procedurales*, lo que quiere decir que el usuario dice al sistema exactamente cómo debe manipular los datos. Otros son *no procedurales*, que significa que el usuario dice qué datos necesita, en lugar de decir cómo deben obtenerse.

En los siguientes puntos se presentan el álgebra relacional y el cálculo relacional, definidos por Codd como la base de los lenguajes relacionales. Se puede decir que el álgebra es un lenguaje procedural, mientras que el cálculo relacional es un lenguaje no procedural.

2.3.3.1 Álgebra Relacional

El álgebra relacional, al igual que el cálculo relacional, tienen como función en el modelo relacional, el de servir como modelos teóricos de la capacidad de recuperación de la información que todo lenguaje de consulta debe poseer como mínimo. El álgebra relacional consiste en un conjunto de operadores que son aplicados a las relaciones produciendo una nueva relación, fue creada por E. CODD como un conjunto de operaciones sobre las relaciones, es un lenguaje de consulta de procedimientos. Existen cinco operaciones fundamentales unión, diferencia, producto cartesiano, proyección, y restricción, se introducen tres operaciones adicionales reunión, intersección y división definidas en términos de las operaciones fundamentales.

Operaciones con conjuntos

Las operaciones fundamentales con conjuntos, son operaciones binarias, estas producen a partir de dos relaciones una tercera.

Unión.

Entrega como resultado, el conjunto de las tuplas contenidas en ambas relaciones que deben tener el mismo número de atributos y cada uno de estos con iguales dominios, su sintaxis: Relación U Relación

Diferencia.

Establece las tuplas contenidas en una relación que no se encuentran en la otra, ambas relaciones deben tener el mismo número de atributos, y estos a su vez con el mismo rango de dominios, su sintaxis: Relación – Relación

Producto Cartesiano.

Establece las posibles combinaciones entre dos tipos de relaciones, su sintaxis: Relación X Relación

Operaciones unarias.

Estas operaciones toman como argumento una relación produciendo una única relación.

Proyección.

Devuelve tuplas unívocas sobre ciertos atributos, su sintaxis: $\Pi_{columnas}$ (Relación)

Restricción.

Selecciona tuplas que cumplen con un predicado determinado, su sintaxis: $\sigma_{predicado}$ (Relación)

Operaciones Adicionales.

Concatenación

La concatenación de dos relaciones R y S obtiene como resultado una relación cuyas tuplas son todas las tuplas de R concatenadas con todas las tuplas de S que en los atributos comunes tienen los mismos valores. Estos atributos comunes aparecen una sola vez en el resultado. Devuelve el conjunto de tuplas contenidas que coinciden con ambas relaciones, ambas

relaciones deben tener el mismo número de atributos, y éstos a su vez con el mismo rango de dominios.

Intersección.

Devuelve el conjunto de tuplas contenidas que coinciden con ambas relaciones, ambas relaciones deben tener el mismo número de atributos, y éstos a su vez con el mismo rango de dominios, su sintaxis: Relación ∩ Relación

División

Trabaja sobre una relación de grado dos con otra de grado uno, donde la relación de grado uno debe tener un dominio semejante con alguno de los atributos de la relación de grado dos. El resultado es determinar las tuplas coincidentes sobre las instancias de las relaciones, su sintaxis: Relación ÷ Relación

2.3.3.2 Cálculo Relacional

El cálculo relacional toma su nombre del cálculo de predicados, que es una rama de la lógica. Existen dos tipos de cálculo relacional, el primero es el *orientado a tuplas*, propuesto por Codd, y el *orientado a dominios*, propuesto por otros autores.

En el cálculo de predicados (lógica de primer orden), un *predicado* es una función con argumentos que puede tomar los valores de verdadero o falso. Cuando los argumentos se sustituyen por valores, la función lleva a una expresión denominada *proposición*, que puede ser verdadera o falsa. Por ejemplo, las frases ‘Juan Carlos es un miembro de la plantilla' y ‘Juan Carlos gana más que Adriana' son proposiciones, ya que se puede determinar si son verdaderas o falsas. En el primer caso, la función ‘es un miembro de la plantilla' tiene un argumento (Juan Carlos) y en el segundo caso, la función ‘gana más que' tiene dos argumentos (Juan Carlos y Adriana).

Si un predicado tiene una variable, como en 'x es un miembro de la plantilla', esta variable debe tener un *rango* asociado. Cuando la variable se sustituye por alguno de los valores de su rango, la proposición puede ser cierta; para otros valores puede ser falsa. Por ejemplo, si el rango de x es el conjunto de todas las personas y se reemplaza x por Juan Carlos, la proposición 'Juan Carlos es un miembro de la plantilla' es cierta. Pero si se reemplaza x por el nombre de una persona que no es miembro de la plantilla, la proposición es falsa.

Cálculo Relacional Orientado a Tuplas

Concepto: Lenguaje de consulta de datos formal que permite expresar preguntas a partir de fórmulas en donde las variables están interpretadas como una variante sobre las tuplas de una relación.

En el cálculo relacional orientado a tuplas, lo que interesa es encontrar tuplas para las que se cumple cierto predicado. El cálculo orientado a tuplas se basa en el uso de *variables tupla*. Una variable tupla es una variable cuyo rango de valores son las tuplas de una relación.

Cálculo Relacional Orientado a Dominios

Este cálculo difiere del cálculo de tuplas en que sus variables varían sobre dominios en lugar de tuplas.

Aunque el cálculo relacional es difícil de entender y de usar, tiene una propiedad muy atractiva: es un lenguaje no procedural. Esto ha hecho que se busquen técnicas no procedurales algo más sencillas, resultando en dos nuevas categorías de lenguajes relacionales: orientados a transformaciones y gráficos.

Los *lenguajes orientados a transformaciones* son lenguajes no procedurales que utilizan relaciones para transformar los datos de entrada en la salida deseada. Estos lenguajes tienen estructuras que son fáciles de utilizar y que permiten expresar lo que se desea en términos de lo que se conoce. Uno de estos lenguajes es SQL (Structured Query Language).

Los *lenguajes gráficos* visualizan en pantalla una fila vacía de cada una de las tablas que indica el usuario. El usuario rellena estas filas con un 'ejemplo' de lo que desea y el sistema devuelve los datos que siguen tal ejemplo. Uno de estos lenguajes es QBE (Query-by-Example).

2.4 Bases Relaciónales

Por lo visto anteriormente se puede decir, que una base de datos relacional es aquella que es percibida por los usuarios como un conjunto de tablas. Es posible considerar dos tipos de almacenamiento de datos en un sistema de información basado en computadora: archivos individuales o elaborar una base de datos. Un sistema que hace uso de los archivos convencionales implica que los datos almacenados lleguen a ser redundantes, la actualización de los archivos lleva más tiempo y existirán problemas de integración de datos, ya que los cambios en un archivo, requieran la modificación de ciertos datos en otros archivos.

Una base de datos es una colección de datos integrada, generalizada y estructurada atendiendo a las relaciones naturales de modo que suministre todos los caminos de acceso necesarios a cada unidad de datos con el objeto de poder atender todas las necesidades de los diferentes usuarios. Esta colección de datos debe reflejar las interrelaciones del mundo real; los datos podrán ser compartidos, deben ser independientes de las aplicaciones y su definición y descripción deben ser únicas para cada tipo de dato. Los procedimientos de actualización y recuperación serán comunes y bien determinados, habrán de ser capaces de conservar la integridad, seguridad y confiabilidad del conjunto de datos.

Así pues, una base de datos es una fuente central de datos significativos que pueden ser compartidos por varios usuarios y por varias aplicaciones y asegura que los datos requeridos estén siempre disponibles, una base de datos permite el mantenimiento preciso y consistente de los mismos.

Las ventajas del almacenamiento en una base de datos, son las siguientes:

Redundancia. Se refiere a la existencia de información repetida o duplicada en diferentes tablas dentro de la base de datos. La redundancia lleva a problemas que se involucran con la integridad y consistencia de los datos. En una base relacional este concepto es mínimo.

Consistencia. Este concepto lleva a la integridad referencial. Una base de datos que se encuentre en inconsistencia puede suministrar información incorrecta o contradictoria.

Integridad. Se refiere no solo a que los datos sean consistentes, sino también, que los valores que poseen los datos sean válidos de acuerdo a las dependencias funcionales entre las tablas de acuerdo a las políticas establecidas para la base de datos.

Seguridad. Un sistema de información es seguro si se puede confiar en que el y su software se comportarán como se espera que lo haga y que la información almacenada en el se mantendrá inalterada y accesible durante el tiempo que sea necesario. La seguridad implica que los usuarios están autorizados para llevar a cabo lo que traten de hacer. Algunos sistemas operativos proporcionan algún nivel de seguridad en el control de acceso a usuarios, sin embargo, esta debe radicar principalmente en el sistema manejador de base de datos (DBMS) o en la aplicación que maneje la base de datos. La seguridad en una base de datos se da en tres niveles:

Primer nivel. Se da a nivel de servidor (LOGIN)

Segundo nivel. Se da a nivel de Base de Datos (USUARIO)

Tercer nivel. Se da a nivel de operaciones y objetos, donde el nivel de operaciones se refiere a los permisos para poder modificar los datos (insertar, borrar, actualizar) y nivel de objetos se refiere a los permisos de los diferentes usuarios para poder hacer uso de las tablas, listas y procedimientos almacenados.

Lo mencionado anteriormente proporciona la ventaja de una base de datos sobre los archivos individuales para el almacenamiento de datos.

Para un sistema basado en computadora de manejo de base de datos, es de gran importancia tener una base de datos diseñada de manera adecuada, lo que permite obtener las características de una base relacional, para esto se realiza el modelado de la base de datos.

El modelo (diseño) de una base de datos se realiza a dos niveles. El primero es el nivel conceptual, en la cual se contempla una estructura abstracta y que no se puede implementar directamente con un SGBD. El segundo es el nivel físico, en el cual la base de datos puede implementar con un SGBD. Detalladamente, las fases del diseño de una base de datos son las siguientes:

1. Descripción en lenguaje natural.
2. Diagrama Entidad-Relación (E-R). También conocido como "diagrama de Chen". Estos diagramas modelan el problema mediante entidades asociadas por relaciones.
3. Elección del modelo de datos (usualmente el relacional).
4. Conversión del diagrama E-R al modelo relacional (tablas).
5. Normalización (eliminar diversos defectos de diseño).
6. Optimización (según criterios de almacenamiento interno, como el espacio en disco y el tiempo medio de acceso).

Las tres primeras fases pertenecen al nivel conceptual del modelo de bases de datos mientras que las tres últimas se relacionan con el nivel físico. La fase de normalización, es la simplificación de las relaciones existentes entre los datos de las aplicaciones y los almacenamientos usados por los usuarios; de tal manera, que se establezcan estructuras de datos de menor tamaño. Es un proceso de conversión de las asociaciones entre las entidades, evitando:

- La redundancia de los datos en un sistema.
- Inconsistencias de los datos como resultado de datos redundantes y actualizaciones parciales.

- Pérdidas no intencionadas de datos debido a que se han borrado otros datos.
- Imposibilidad de adicionar datos en la base de datos debido a la ausencia de otros datos.

Se emplean tres pasos para simplificación de los datos, llamados formas normales (FN).

Primera forma normal (FN1), una entidad esta en primera forma normal si cada uno de los campos contiene un único valor para el registro determinado, es decir, elimina los grupos de datos repetidos e identifica la llave primaria.

Segunda forma normal (FN2), compara todos y cada uno de los campos de las entidades con la llave primaria, si todos los campos dependen directamente de la llave primaria se esta en segunda forma normal, esta forma se aplica para asegurar que todos los atributos no llave sean dependientes de la llave primaria.

Tercera forma normal (FN3) elimina cualquier dependencia transitoria, es decir que los campos de la tabla dependen únicamente de la llave primaria y no dependen de ningún otro atributo no llave de la tabla.

Manejador de Bases de Datos

En una base de datos una parte de gran importancia es el DBMS, esto es, entre la base de datos física y los usuarios existe un nivel de programas denominado “Manejador de Base de Datos” (MDB) o “Sistema Administrador de Base de Datos” (DBMS, Database Manager System).

Un DBMS es el conjunto de programas que maneja todo acceso a la base de datos, permite la definición, manipulación y control de una base de datos, facilitan la integridad, seguridad y concurrencia de los datos.

Los componentes de un Sistema Administrador de Base de Datos son:

- DLL (Lenguaje de Definición de Datos). Permite la definición o descripción de los objetos de la base de datos, se utiliza para crear, alterar o borrar tablas, vistas, restricciones de integridad (llaves primarias y foráneas), etc.

- DML (Lenguaje de Manipulación de Datos). Permite el procesamiento de los objetos de la base de datos, se utiliza para leer, modificar, borrar o agregar tuplas a las relaciones.

- DCL (Lenguaje de Control de Datos). Permite la definición de los privilegios del control de acceso y edición a los elementos que componen la base de datos.

- DD (Diccionario de Datos). El contenido del diccionario de datos es la definición de los objetos contenidos en la base de datos.

El objetivo primordial de un sistema manejador base de datos es proporcionar un ambiente que sea a la vez conveniente y eficiente para ser utilizado al extraer, almacenar y manipular información de la base de datos. Todas las peticiones de acceso a la base, se manejan centralizadamente por medio del DBMS, por lo que este paquete funciona como interfase entre los usuarios y la base de datos.

Las funciones principales de un DBMS son:

- Crear y organizar la Base de datos.

- Establecer y mantener las trayectorias de acceso a la base de datos de tal forma que los datos puedan ser obtenidos rápidamente.

- Manejar los datos de acuerdo a las peticiones de los usuarios.

- Registrar el uso de las bases de datos.

- Interacción con el manejador de archivos. Esto a través de las sentencias en DML al comando del sistema de archivos. Así el Manejador de base de datos es el responsable del verdadero almacenamiento de los datos.

- Respaldo y recuperación. Consiste en contar con mecanismos implantados que permitan la recuperación fácilmente de los datos en caso de ocurrir fallas en el sistema de base de datos.

- Control de concurrencia. Consiste en controlar la interacción entre los usuarios concurrentes para no afectar la inconsistencia de los datos.

- Seguridad e integridad. Consiste en contar con mecanismos que permitan el control de la consistencia de los datos evitando que estos se vean perjudicados por cambios no autorizados o previstos.

El DBMS es conocido también como Gestor de Base de datos.

En sí, un sistema manejador de base de datos es el corazón de la base de datos ya que se encarga del control total de los posibles aspectos que la puedan afectar.

Los manejadores de bases de datos disponibles en Radio UNAM para el desarrollo del sistema de la fonoteca son:

SQL Server 2000

SQL fue diseñado por IBM, se ha convertido en la norma de facto, debido a que prácticamente todos los sistemas relacionales, implementan alguna versión de SQL.

Su arquitectura permite una excelente escalabilidad sobre servidores SMP (multiprocesamiento simétrico) con memoria adicional, su desempeño es relativo a su necesidad de memoria, proporciona bloques de construcción simples que pueden ejecutar de manera confiable y eficiente transacciones a través de redes distribuidas generalizadas.

Características de SQL Server 2000:

- Permite tener las bases de datos INTEGRADAS en la misma ventana y realizar cualquier modificación sobre ellas.

- Permite olvidarse de los archivos que forman la base de datos.

- Soporte para OLDB.
- Ejecución paralela de consultas.
- Consultas distribuidas.
- Agrupación y gestión de fallas.
- Administración de permisos a nivel servidor, seguridad en tablas, seguridad en los procedimientos almacenados.

Los requerimientos mínimos se indican a continuación:

- Procesador Pentium 166 MHz o superior.
- 32 MB de memoria RAM mínimo.
- 65 MB de espacio libre en disco duro para instalación mínima.
- Permite ser ejecutado en plataformas Windows 95 y superior.

SQL Server proporciona varias ventajas de considerable importancia como lo son:

- Asignación dinámica de recursos. La asignación dinámica de recursos del SQL Server es una característica muy útil, permite la escalabilidad del uso del disco y memoria para acomodarse a las necesidades de la base de datos en cada momento. Esta flexibilidad permite un mejor rendimiento y simplifica la administración del software.
- El soporte para la plataforma Win9x aumenta significativamente la base de aplicaciones posibles para el SQL Server.
- El Analizador Gráfico de Consultas. Permite construir procedimientos guardados y ejecutar consultas interactivas, además, es posible visualizar gráficamente los pasos que el procesador de consultas usa para ejecutar la consulta.

- Los Servicios de Transformación de Datos (DTS). Facilita el almacenamiento de datos, simplifica la importación y la exportación de datos entre dos bases compatibles con OLE DB, además, genera scripts Visual Basic (VBScript) que se puede ejecutar desde el WSH (Windows Scripting Host) u otros entornos COM (Component Object Model).

Desventajas de SQL Server:

- La instalación y operación requiere, mínimo, del Internet Explorer (IE) 4.0.
- La migración requiere un reinicio de la base de datos.

Microsoft ACCESS 2000

Access es un sistema de manejo de bases de datos o DBMS por sus siglas en inglés (Data Base Management System). Access está basado en el modelo Relacional de Bases de datos por lo que se le puede considerar un manejador de Bases de Datos Relacionales RDBMS (Relational Data Base Management System).

Características de Acces

- Mejora del rendimiento y menores requerimientos de memoria.
- Facilidad de construir soluciones Internet e Intranet.
- Integración y consistencia con Microsoft Office 97.
- Mejora del entorno de desarrollo.
- Replicación, de manera que usuarios desconectados y oficinas remotas pueden trabajar en copias locales de los datos.

Soporta las últimas mejoras del sistema operativo, incluyendo los controles ActiveX, automatización ActiveX, nombres largos de archivos, etc.

Publicación en Web, de manera que parte o toda la aplicación puede ser transferida a soluciones Internet/Intranet.

Control de código fuente integrado el cual ayuda a usuarios simples y equipos de desarrollo.

Ventajas de Access.

Access es un manejador de bases de datos para aplicaciones grandes, a continuación se enlistan algunas de estas ventajas:

- Tiene integración completa a SQL Server, esto permite migrar fácilmente a esta plataforma si por necesidades técnicas se requiriese de un manejador más robusto o que soporte mayor número de registros.
- Access esta totalmente preparado para integración de datos al WEB.
- Contiene herramientas para desarrollo de formularios que permiten usarlo tanto de Back end como de Front end.
- Admite sentencias SQL complejas.
- Permite manejar BD de hasta un Tera Byte de información.
- Permite realizar reportes de forma directa.
- Existe la posibilidad de migrar en forma transparente a casi cualquier otro manejador de BD.
- Los Front Ends diseñados en Access se pueden migrar de forma transparente a Visual Basic.

MySQL

Es un Sistema de Administración de Base de Datos Relacional (RDBMS-Relational DataBase Management System). Esta liberado bajo GNU GPL para ser código libre.

Características

- Su principal objetivo de diseño fue la velocidad para las consultas.

- Consume pocos recursos, tanto de CPU como de memoria.
- Licencia GPL a partir de la versión 3.23.19.

Requerimientos para MySQL:

- Procesador Pentium (o superior) 450 MHz o superior.
- 128 MB en memoria RAM.
- 100 MB de espacio libre en disco duro.
- Servidor Web por ejemplo: Microsoft IIS Server 4.0 o superior.

Como principales ventajas se observan:

- Mayor rendimiento. Mayor velocidad tanto al conectar con el servidor como al proporcionar los resultados de las consultas.
- Mejores utilidades de administración (backup, recuperación de errores, etc).
- Mejor integración con PHP.
- No hay límites en el tamaño de los registros.
- Mejor control de acceso, en el sentido de qué usuarios tienen acceso a qué tablas y con qué permisos.
- MySQL se comporta mejor que Postgres a la hora de modificar o añadir campos a una tabla "en caliente".

Las desventajas más significativas son:

- No soporta transacciones, "roll-backs" ni subselects.
- No considera las claves ajenas. Ignora la integridad referencial, dejándola en manos del programador de la aplicación.

Un DBMS tiene las funciones mencionadas anteriormente, las cuales realiza mediante un lenguaje de consulta, con el que pueden comunicarse los usuarios con el DBMS, a continuación se describe el lenguaje más importante para el manejo de las bases de datos relacionales, el lenguaje es el SQL.

Lenguaje de Consulta SQL.

El álgebra y el cálculo relacionales, sirven como patrón de medida para poder evaluar el poder de recuperación de la información de cualquier otro lenguaje. Sin embargo, estos dos lenguajes no se manejan como herramientas prácticas de acceso a los datos debido a que su sintaxis es estilizada y concisa y resulta natural únicamente a usuarios de orientación matemática, aunque en teoría son “completos ” en cuanto a la extracción de datos, ambos lenguajes carecen de construcciones que permitan resultados a partir de valores existentes en la base de datos, por ejemplo, consultas como “cuántos usuarios manejan Linux” ó “cuál es el examen más reciente de sistemas” no son expresables en estos lenguajes, otro punto es que tanto el álgebra como el cálculo no incluyen funciones de actualización de datos, necesarias en cualquier lenguaje práctico. En la práctica, los lenguajes de consulta incorporan funciones administrativas además de las de consulta y actualización.

El lenguaje SQL que originalmente fue diseñado por IBM e incorporado en varios de sus productos, se ha convertido en la norma de facto, debido a que prácticamente todos los sistemas relacionales, implementan alguna versión de SQL. Las organizaciones ANSI e ISO de normalización, han desarrollado sucesivas definiciones de un SQL estándar. Este lenguaje comercial utiliza una combinación del álgebra relacional y construcciones del cálculo relacional, además de ser un lenguaje de consulta, presenta la capacidad de definir la estructura de los datos, para modificar datos en la base, y para especificar las limitantes de seguridad. SQL, bajo su nombre original de SEQUEL (Structered English Query Languaje), fue diseñado en la década del 70 como parte del proyecto de investigación Sistema R de IBM.

CAPÍTULO II

DESCRIPCION DEL LENGUAJE.

Definición de datos.

Para la definición de datos SQL utiliza los verbos:

create: para crear un esquema de relación.

alter: modificar un esquema de relación.

drop: eliminar un esquema de relación.

Consulta de datos.

Para hacer la consulta de datos SQL utiliza el verbo:

select: prefija todas las operaciones de consulta.

La forma general de esta operación es la siguiente:

```
select  [distinct] atributo (s)
from relación(es)
[where fórmula]
[group  by atributo(s) [having fórmula]]
[order by atributo(s)]
```

La cláusula from indica a que relación (es) se aplica select.

La cláusula distinct elimina las tuplas repetidas en el resultado de la selección.

La cláusula where (donde) restringe mediante una fórmula el conjunto de tuplas que se desea extraer, puede utilizar condiciones complejas formadas con los operadores and, or y not, y las comparadores =, <,>,<=,>=, a los cuales se denominan predicados de comparación, entre los que se incluyen between y su negación no between. Otros operadores son like y null; null es utilizado para determinar si un componente de una tupla es nulo y su sintaxis es: nombre de atributo is null. El predicado like es utilizado con los caracteres especiales “%” y ” _” que son

meta símbolos donde "%" representa cero o más caracteres, mientras que "_" representa exactamente un carácter. Estos meta símbolos son utilizados en la búsqueda de patrones en los datos donde su forma general es: nombre del atributo like "patrón de caracteres"

La cláusula order by (ordenar por) ofrece cierto control sobre el orden en que se van a mostrar las tuplas de una relación.

La cláusula group by (agrupar por) permite formar grupos. Las tuplas que tengan el mismo valor para el atributo se colocan en un grupo. El SQL incluye funciones para realizar los calculos:

- Mínimo: min
- Máximo: max
- Total: sum
- Número de tuplas: count.

Actualización de datos.

SQL provee de tres sentencias que permiten modificar el contenido de una relación o relaciones: insert, delete y update, su sintaxis es:

```
insert into relación [campo(s)]  values (constante(s));
update relación set {campo= exp}[where condición];
delate relación [where condición];
```

Sentencia insert (inserte) permite insertar registros a una relación existente.

Sentencia update (modifique) permite modificar registros dentro de una relación existente.

Sentencia delete (suprime) permite eliminar tuplas de una relación.

Definición de vistas.

El término vista se refiere a una relación que no es parte del modelo conceptual y que se hace

visible para el usuario en forma de “relación virtual”. A diferencia de una tabla real, una vista no existe en la base de datos como un conjunto almacenado de valores. Las vistas son parte importante de SQL, permiten acomodar el aspecto de una base de datos de modo que diferentes usuarios la vean desde diferentes perspectivas, restringen acceso a datos y simplifican el acceso a la base de datos.

Una vista se define en SQL empleando create view (crear vista). Para definir una vista se le debe dar nombre y especificar la consulta que calcula la vista, su sintaxis es:

```
create view nombre_vista as <expresión de consulta>
```

Restricción de datos

Algo importante de considerar con respecto en la creación de las bases de datos es la integridad de los datos que contiene. Existen dos formas de mantener la integridad, una es aplicar las restricciones en los programas de aplicación, la otra forma es mediante declaraciones al SGBD. Algunas restricciones son las siguientes.

Restricciones aplicables a atributos.

- Not null especifica que no es posible ingresar un valor nulo para el atributo.
- Unique especifica que los valores del atributo son únicos, no puede existir dos o más filas con el mismo valor para ese atributo.
- Llave Primaria. Declara un atributo o conjunto de atributos como clave primaria de una tabla.

Integridad Referencial.

Este concepto se relaciona con llaves foráneas (foreing key), el cual ya se explico anteriormente. El lenguaje SQL a manera de resumen permite:

- SQL se utiliza para recuperar datos de la base de datos.
- SQL permite actualizar la base de datos.

- SQL permite controlar el acceso a la base de datos.
- SQL se utiliza para crear la base de datos.

Actualmente las redes de computadoras, han permitido el desarrollo de sistemas de información de mayor alcancé con el fin de compartir recursos para los usuarios, ejemplo de ellos son: correo electrónico, servicios de Internet, servicios de impresión, servicios de base de datos, entre otros.

2.5 Redes y Comunicación

Una red es un conjunto de computadoras conectadas por medios físicos y lógicos con el fin de transferir e intercambiar datos entre computadoras, para realizar una actividad de forma eficiente y eficaz.

Clasificación de las redes

Las redes de computadoras son clasificadas en base a varios criterios, siendo el más importante la cobertura geográfica, por la que se clasifican en: Local Area Network (LAN) “Redes de Área Local”, Metropolitan Area Network (MAN) “Redes de Área Metropolitana” y Wide Area Network (WAN) “Redes de Área Amplia o Extendida”. Por su topología las redes locales se clasifican en: bus, estrella, anillo y malla.

Redes de Área Local.

Estas redes son propiedad de la organización que las utiliza dentro de un espacio determinado, ya sea una oficina, edificio o terreno, transmitiendo datos entre estaciones de usuario y computadoras para compartir recursos e intercambiar información, las velocidades típicas varían desde 10 a 100 Mbps.

Redes de Área Metropolitana.

Es una red local muy grande que cubre una ciudad entera, suministrando el transporte de datos a grandes velocidades utilizando fibra óptica. Típicamente una MAN conectará LANs de más baja velocidad a través de una ciudad o región, solucionando las limitaciones de ancho de banda.

Redes de Área Amplia o Extendida.

Con el fin de interconectar computadoras en áreas geográficamente muy alejadas o redes LAN y MAN, se requiere contar con mecanismos de acceso especiales que caen dentro del concepto internetwork o “internet”.

Topologías de Red.

Establece la forma en cuanto a conectividad física de los equipos computacionales a la red. Para establecer la topología de una red se tiene como objetivos proporcionar la máxima fiabilidad en el tráfico de la información, encaminar el tráfico utilizando la vía de coste mínimo entre el transmisor y el receptor y proporcionar al usuario el rendimiento óptimo y tiempo de respuesta mínimo.

Topología Bus (línea).

En esta topología habitualmente solo existe un canal único de comunicación al que se conectan todos los dispositivos de la red, esta canal de transmisión (bus) permite que cada equipo pueda difundir a todos los demás equipos la información, aunque solo es procesado por aquella a la que va dirigido, los extremos del canal (cable) no están interconectados sino simplemente finalizados con un terminador, el terminador elimina automáticamente la señal de los extremos.

Al tener un canal compartido existen dos problemas que deben ser resueltos: uno es que varios dispositivos intenten transmitir al mismo tiempo sobre el bus, produciéndose una colisión (se mezclan los mensajes y el resultado es incomprensible), el otro es cuando una estación esté transmitiendo continuamente y monopolice la red. Para evitar eso, los mensajes se transmiten en paquetes de datos más pequeños, haciendo una pausa entre los mismos para dar la oportunidad de transmitir a otras estaciones.

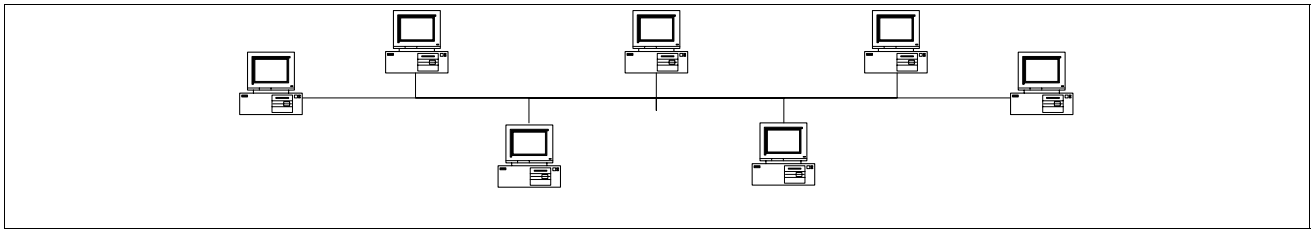


Figura 2.5.1 Topología Bus

Topología en Estrella.

La red se une en un único punto, normalmente con control centralizado, como un concentrador de cable, todas las estaciones se conectan al concentrador y las señales son distribuidas a todas las estaciones. Teniendo como ventaja de que una conexión interrumpida no afecta al resto de la red.

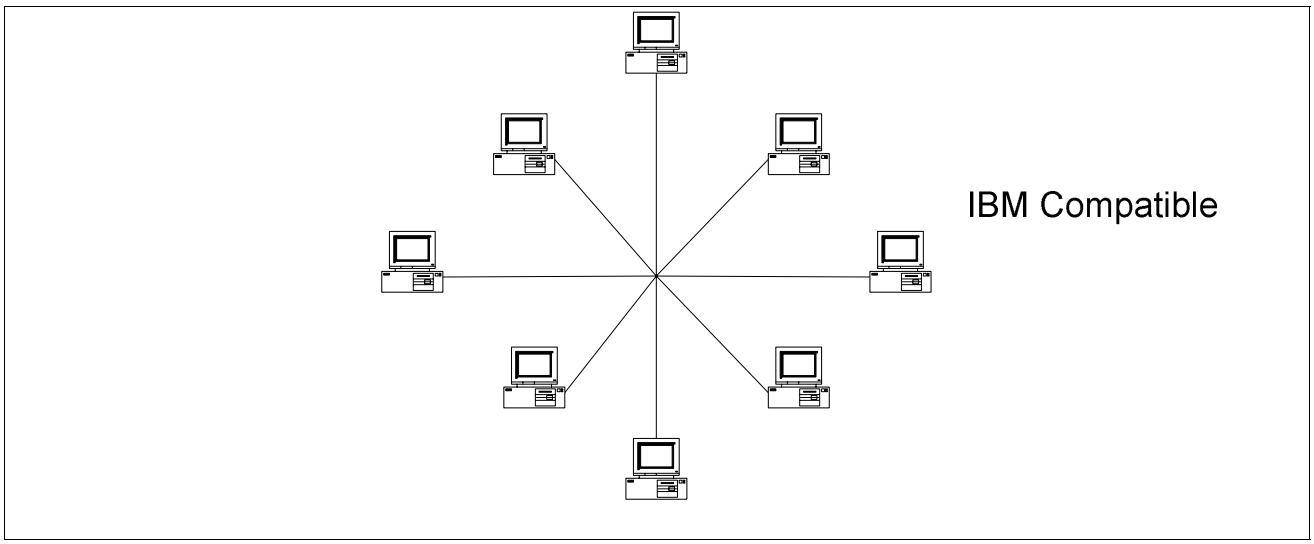


Figura 2.5.2 Topología Estrella

Topología en Anillo (ring).

Recibe su nombre del aspecto circular del flujo de datos, que generalmente va en una sola dirección, las tareas que realiza cada componente son aceptar los datos y enviarlos al siguiente componente en el anillo, siendo muy atractiva debido a que los cuellos de botella son muy raros.

El principal inconveniente de esta topología es que solo tiene un solo canal que une a todos los componentes del anillo, si falla el canal falla toda la red.

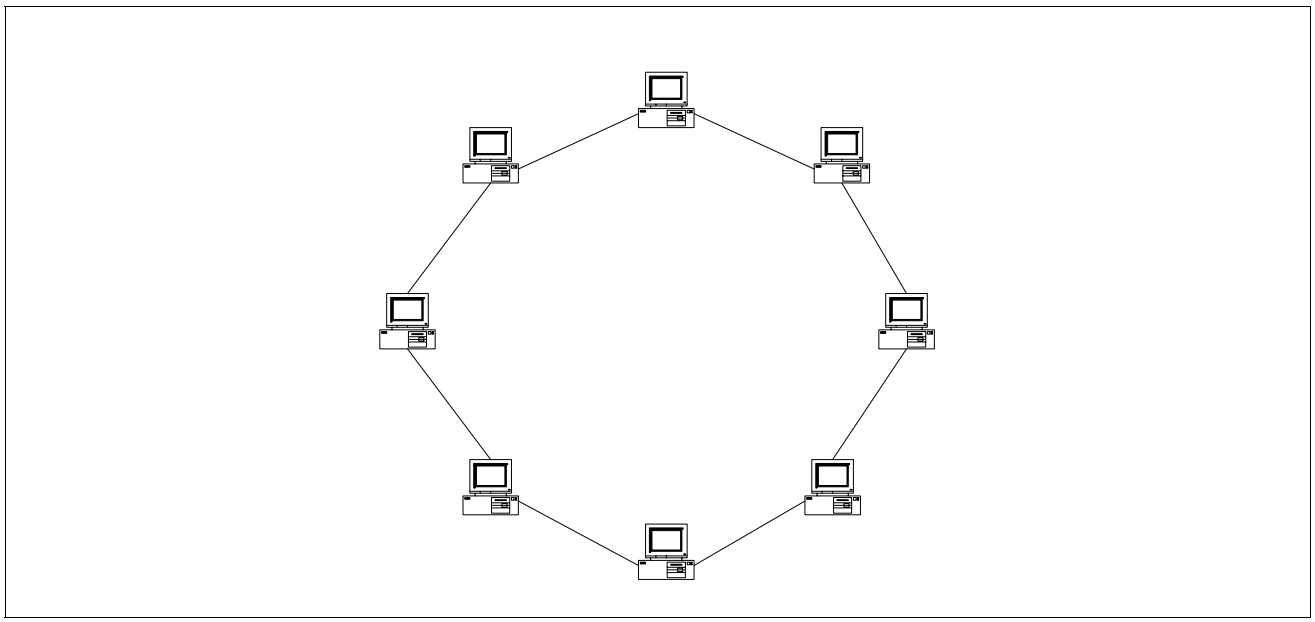


Figura 2.5.3 Topología Anillo

Topología en malla.

Su principal atractivo es su relativa inmunidad a fallos y cuellos de botella Por su multiplicidad de caminos es posible encaminar el tráfico evitando componentes que fallan o nodos ocupados.

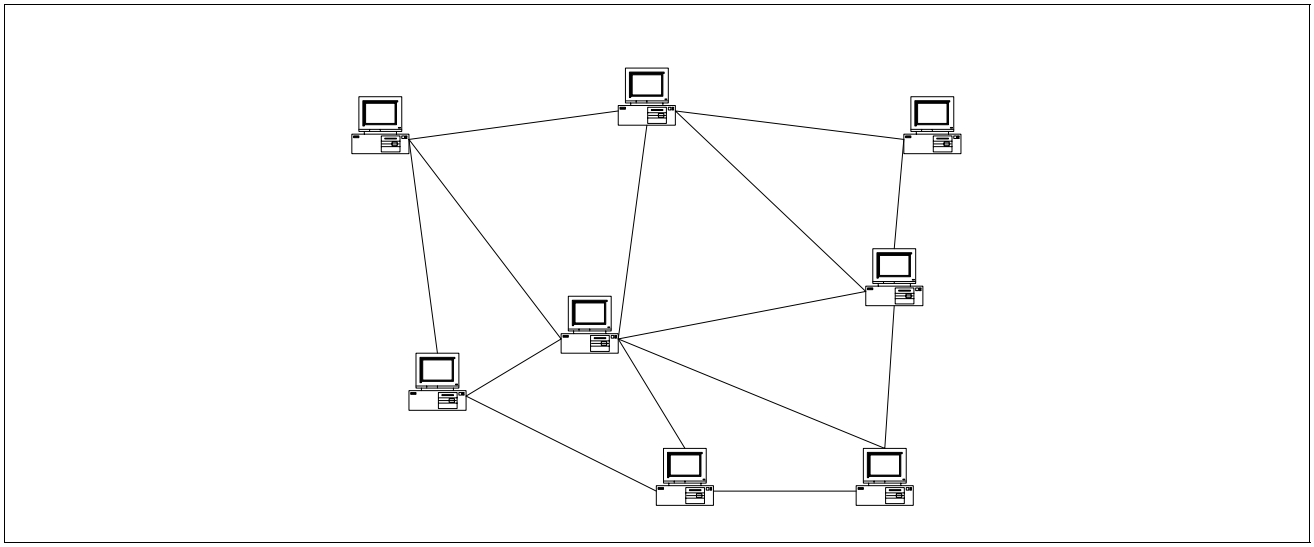


Figura 2.5.4 Topología Malla

PROTOCOLOS

Un protocolo de red es como un lenguaje para la comunicación de información. Son las reglas y procedimientos que se utilizan en una red para comunicarse entre los nodos que tienen acceso al sistema de cable. Los protocolos pueden describir detalles de bajo nivel de las interfaces de máquina a máquina, estos definen la forma en que se transmiten las señales por cable, o del intercambio entre programas de aplicaciones.

Los protocolos más comúnmente utilizados en las redes son Ethernet y Token Ring: IBM Compati

El protocolo Token ring se basa en la topología de anillo. En este protocolo cuando ninguna computadora desea transmitir, circula continuamente sobre el anillo un mensaje especial (sin dirección destino) denominado token. Cuando una computadora desea transmitir espera a recibir el token, lo retira de la red e inicia su transmisión. Ninguna otra computadora puede ahora transmitir ya que el token no se encuentra en el anillo.

Cuando la computadora que tomo el token termina su transmisión coloca un nuevo token en el anillo para que otras computadoras puedan transmitir.

CAPÍTULO II

Ethernet es un protocolo sencillo de bajo nivel que utiliza la topología de bus lineal. El acceso al medio se controla con un sistema conocido como CSMA/CD (Carrier Sense Multiple Access with Collision Detection, “Acceso Múltiple por detección de portadora/detección de colisiones”: En éste sistema, las estaciones de trabajo conectadas a la red escuchan un tono de la portadora en el cable y envían la información en el momento que otras estaciones no están transmitiendo. El acceso múltiple significa que muchas estaciones comparten el mismo cable. Si dos o más estaciones detectan que la red está preparada para transmitir, pueden intentar acceder a esta de forma simultánea, provocando colisiones. Cada estación de trabajo debe entonces abandonar la operación y esperar una cierta cantidad de tiempo antes de empezar a retransmitir. El aumento del número de estaciones de la red incrementa la colisión, pero se puede dividir en segmentos, para evitar la congestión.

La pila TCP/IP se llama así por dos de sus protocolos más importantes: TCP (Transmission Control Protocol) “Protocolo de Control de Transmisión” y de IP(Internet Protocol) “Protocolo de Internet”. Es un protocolo estándar de la industria que permite comunicación en diferentes ámbitos de redes.

TCP/IP utiliza cuatro capas en su modelo de comunicación para transmitir datos de un lugar a otro, las cuatro capas del modelo son:

Aplicación. Es la capa más alta de TCP/IP, todas las utilidades y aplicaciones están contenidas en esta capa y son utilizadas para tener acceso a la red.

Transporte. Tiene la capacidad para ordenar y garantizar la comunicación entre computadoras y pasar los datos de la capa de aplicación hacia la capa de internet, también especifica un identificador único de aplicación para el cual los datos son enviados.

Internet. Es responsable de direccionar, empaquetar y rutear los datos para ser transmitidos.

Interfase de red. Es la responsable de poner y recibir los datos en el medio de la red. Esta capa contiene dispositivos físicos de la red tal como cables y adaptadores de red.

El protocolo IP proporciona tres definiciones importantes: define la unidad básica para la transferencia de datos utilizada a través de una red de redes, es decir, especifica el formato de

todos los datos; el software IP realiza la función de ruteo, seleccionando la ruta por la que los datos serán enviados y además de aportar especificaciones para el formato de los datos y el ruteo. Se trata de un sistema de entrega de paquetes (llamados datagramas IP).

El datagrama IP (figura 2.5.5) es la unidad básica de transferencia de datos entre el origen y el destino. Se divide en áreas de encabezado y datos, el encabezado contiene la dirección de la fuente y del destino, también contiene un campo de tipo que identifica el contenido del datagrama. Viaja en el campo de datos de las tramas físicas de las distintas redes que va atravesando. Cada vez que un datagrama tiene que atravesar un router, el datagrama saldrá de la trama física de la red que abandona y se acomodará en el campo de datos de una trama física de la siguiente red. Este mecanismo permite que un mismo datagrama IP pueda atravesar redes distintas: enlaces punto a punto, redes Ethernet, redes Token Ring, etc. El propio datagrama IP tiene también un campo de datos: será aquí donde viajen los paquetes de las capas superiores.



Figura 2.5.5 Datagrama IP

Número de Versión. Es un campo de 4 bits, contiene el número de versión IP utilizado el software del protocolo, se emplea para que el software IP del receptor sepa como se debe descifrar el reste del encabezado.

Longitud del encabezado. Campo de 4 bits con el valor de la longitud de la cabecera del datagrama, la longitud se mide en palabras de 32 bits.

Tipo de servicio. Es de 8 bits, indica al IP acerca de cómo procesar el datagrama de manera apropiada.

Longitud del paquete (longitud del datagrama). Especifica la longitud total del datagrama, incluyendo el encabezado en bytes, la longitud máxima de un datagrama es de 65535 bytes.

Identificador. Contiene un número que es un identificador único que se requiere a la hora de reensamblar mensajes fragmentados. Se utiliza junto con la dirección del receptor para identificar el fragmento.

Banderas. Es un campo de 3 bits, el primero no es utilizado por el protocolo, los bits restantes están dedicados a DF (Don`t Fregment, "No Fragmentar") y MF (More Regment "Más Fragmentos"), los cuales manejan los datagramas cuando se fragmentan, si DF se fija en 1 el datagrama no puede fragmentarse, si MF se fija en 1 al datagrama actual lo siguen más paquetes, los cuales deben reensamblarse para crear el mensaje completo.

Desplazamiento de Fragmentación. Este registro depende de la bandera MF, si esta bandera es fijada en 1 el registro de desplazamiento de fragmentación contiene la posición en el mensaje completo del submensaje contenido dentro del datagrama actual.

Tiempo de Vida. (TTL). Da el tiempo en segundos que un datagrama puede permanecer en la red antes de que se deseche, por lo general se fija en 15 o 30 segundos.

Protocolo de Transporte. Contiene el número de identificación del protocolo de transporte al que fue entregado el paquete. Los dos protocolos más importantes son el ICMP "Protocolo Internet de Mensajes de Control"y el TCP "Protocolo de Control de Transmisión".

Checksum de la cabecera. Se utiliza para detectar distorsiones en la cabecera.

Dirección de Envío y de Destino. Contiene la dirección IP de 32 bits de los dispositivos de envío y de destino, se establecen cuando se crea el datagrama y no se modifican durante toda la vida del datagrama.

Opciones. Se emplea para identificar diversos servicios adicionales, este campo no se utiliza en todos los datagramas.

Datos. Contiene los datos de usuario.

Protocolo TCP

TCP es un protocolo orientado a conexión, mantiene información del estado de cada cadena de datos de usuario que circula por él, es responsable de la transferencia de datos entre extremos de la red o redes hasta la aplicación del usuario receptor. Proporciona servicios a las aplicaciones como: la recuperación de errores, control de flujo y fiabilidad.

TCP se puede caracterizar por los siguientes servicios que suministra a las aplicaciones que lo usan:

- Transferencia de datos a través de un canal: Desde el punto de vista de la aplicación, TCP transfiere un flujo continuo de bytes a través de Internet. La aplicación no ha de preocuparse de dividir los datos en bloques o en datagramas.

- Fiabilidad: TCP asigna un número de secuencia a cada byte transmitido, y espera un reconocimiento afirmativo (ACK) del TCP receptor. Si el ACK no se recibe dentro de un intervalo de tiempo, los datos se retransmiten. El TCP receptor utiliza los números de secuencia para organizar los segmentos cuando llegan fuera de orden, así como para eliminar segmentos duplicados.

- Control de flujo: El TCP receptor, al enviar un ACK al emisor, indica también el número de bytes que puede recibir aún, sin que se produzca sobrecarga y desbordamiento de sus buffers internos. Este valor se envía en el ACK en la forma del número de secuencia más elevado que se puede recibir sin problemas.

- Multiplexación: Se consigue usando puertos, al igual que en UDP.

- Conexiones lógicas: La fiabilidad y el control de flujo requieren que TCP inicialice y mantenga cierta información de estado para cada canal.

- Full Duplex: TCP garantiza la concurrencia de los flujos de datos en ambos sentidos e la conexión.

Unidades de Datos de Protocolo TCP

El formato de las unidades de datos de protocolo TCP se muestra en la figura 2.5.6

<table>
<tr><td colspan="9">Puerto Fuente</td><td colspan="8">Puerto Destino</td></tr>
<tr><td colspan="17">Número de Secuencia</td></tr>
<tr><td colspan="17">Número de Aceptación</td></tr>
<tr><td colspan="3">Desplaza miento de Datos</td><td colspan="3">Reservado</td><td>U R G</td><td>A C K</td><td>P S H</td><td>R S T</td><td>S Y N</td><td>F I N</td><td colspan="8">Ventana</td></tr>
<tr><td colspan="9">Checksum</td><td colspan="8">Puntero de Urgente</td></tr>
<tr><td colspan="9">Opciones</td><td colspan="8">Relleno</td></tr>
<tr><td colspan="17">Datos</td></tr>
</table>

Figura 2.5.6 Unidades de Datos de Protocolo TCP

Puerto Fuente y Destino. Son campos de 16 bits identifican al usuario TCP local y de la máquina remota respectivamente.

Número de Secuencia. Indica la posición del bloque actual en el mensaje total, también se emplea para proporcionar el número de secuencia de envío inicial.

Número de Aceptación. Indica el siguiente número de secuencia esperado.

Desplazamiento de Datos. Este campo se usa para identificar el inicio del campo de datos.

Reservado. Consta de 6 bits que deben valer cero, estos bits están para usos futuros.

Los seis bits siguientes se denominan indicadores, se utilizan para especificar ciertos servicios que se pueden emplear durante la sesión:

URG si está activa (valor de 1) indica que el campo del señalador urgente es significativo.

ACK si está activo indica que el campo de aceptación es significativo.

PSH significa que la función Push va a utilizarse en el módulo.

RST indica que la conexión debe reiniciarse.

SYN indica que los números de secuencia deben sincronizarse.

FIN si está activa indica que el remitente no tiene más datos para enviar.

Ventana. Es un número que indica cuántos bloques de datos puede aceptar la máquina receptora.

El protocolo TCP/IP es el más empleado en las comunicaciones de redes de computadoras, y es utilizado por la gran mayoría de aplicaciones basadas en la arquitectura cliente/servidor.

Arquitectura Cliente/Servidor

La tecnología cliente/servidor es el procesamiento cooperativo de la información por medio de un conjunto de procesadores, en el cual múltiples clientes, distribuidos geográficamente, solicitan requerimientos a uno o más servidores centrales. Desde el punto de vista funcional, se puede definir la computación cliente/servidor como una arquitectura distribuida que permite a los usuarios finales obtener acceso a la información en forma transparente aun en entornos multiplataforma.

En una estructura cliente servidor la computadora que reside por encima de otra se llama servidor y las computadoras que se encuentran debajo de esta son llamadas clientes.

En esta estructura los clientes solicitan servicios que son proporcionados por el servidor, existiendo cuatro tipos de implementaciones distintas:

- Servidores de archivo. El cliente solicita al servidor registros específicos de un archivo.
- Servidores de Bases de Datos. El cliente envía solicitudes en lenguaje de consulta estructurado (SQL) al servidor.
- Servidores de Transacciones. El cliente envía una solicitud la cual emboca procedimientos remotos en el servidor. Se produce la transacción cuando la solicitud ejecuta procedimientos remotos y el resultado es devuelto al cliente.

- Servidores de grupos de trabajo. Este tipo de servidor proporciona un conjunto de aplicaciones que hacen posible la comunicación entre clientes mediante el uso de texto, imágenes, correos electrónicos, vídeo y otras representaciones.

Características de la arquitectura cliente/servidor:

- El Cliente es responsable de la parte lógica y de presentar la información al usuario.
- El Servidor distribuye los recursos disponibles (tales como procesadores, memoria, operaciones de disco, etc.) entre las múltiples peticiones del Cliente. Estos se ve con claridad en la figura 2.5.7.

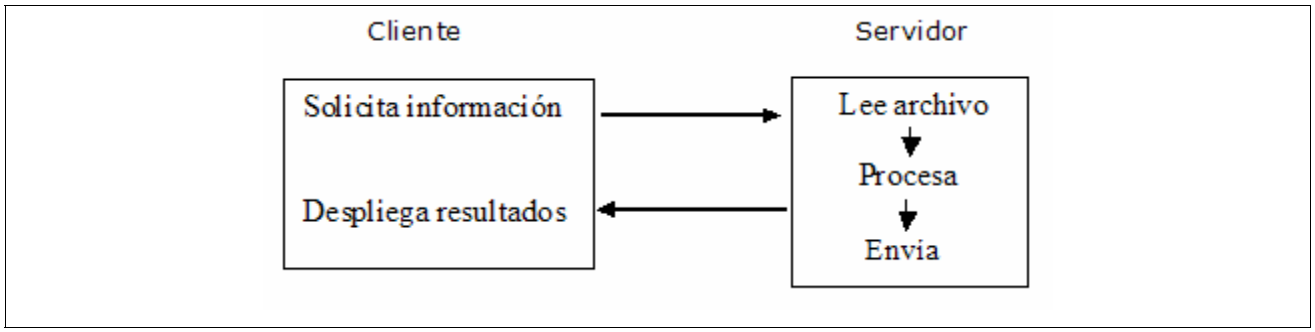


Figura 2.5.7 Arquitectura Cliente/Servidor

La programación Cliente/Servidor se utiliza cuando se quiere realizar aplicaciones que utilicen redes y que comuniquen entre sí a varios equipos de cómputo. Ambas partes de la aplicación se comunican utilizando algún protocolo de red.

El ambiente Cliente/Servidor permite una estructuración, para que los recursos de cómputo se utilicen de la mejor manera posible, con lo cual se reducen los costos de manera considerable, para lograr esto, se deben llevar a cabo ciertas medidas, las cuales se mencionan a continuación:

1. Separación de Tareas: Las funciones de cómputo se pueden sincronizar con los recursos de tal manera que el Servidor procese sólo las que son más complejas o que se requieren operaciones intensas, liberando al Cliente de esa labor, dándole la oportunidad de ejecutar otras tareas simultáneas. Con esta configuración puede integrarse una estrategia que equilibre la combinación y costo de los dispositivos de cómputo.

2. Comportamiento de Periféricos: La combinación de estos dispositivos, también puede realizarse al hacer que sus características coincidan con las funciones que se desea ejecutar y como la arquitectura C/S es abierta, permite la incorporación de un dispositivo especializado para realizar tareas muy específicas.

3. Comportamiento de Herramientas: Los Clientes pueden compartir diversas aplicaciones con la gran ventaja de que éstas pueden ser controladas de manera centralizada por un mismo Administrador. Con esta ventaja, se elimina la necesidad de tener que instalar, configurar y dar de alta la aplicación en cada Cliente cada vez que surja una nueva versión de software.

4. Acceso a la Información: A pesar de que una empresa se encuentre geográficamente separada, la arquitectura C/S permite el acceso a la información a nivel del Servidor de manera transparente, siempre y cuando existan estándares comunes.

En esta arquitectura, el Cliente maneja la interfase del usuario, construye los requerimientos del Servidor y controla la presentación de resultados, incluso puede contener la lógica de la aplicación. Cuando el Servidor recibe una solicitud de servicio, realiza el proceso, regresa el resultado al Cliente y provee el servicio de administración de datos. Tanto el Cliente como el Servidor pueden ejecutar el trabajo, así como la lógica de la aplicación, el Servidor se encarga de ejecutar todos aquellos procesos que para el Cliente podrían representar mucho tiempo de proceso, de tal forma que el Servidor se encarga de realizar el trabajo pesado y el Cliente se dedica a presentar los resultados a los usuarios finales.

Tanto el Cliente como el Servidor son computadoras que tienen un sistema operativo que les permite administrar los dispositivos con los que cuentan y así realizar diversas tareas incluidas las de comunicación en la red.

2.6 Sistemas Operativos

El sistema operativo, es el instrumento indispensable para hacer de la computadora un objeto útil. Un Sistema Operativo es el encargado de brindar al usuario una forma amigable y sencilla de operar, interpretar, codificar y emitir las ordenes al procesador central para que este realice las tareas necesarias y especificas para completar una orden.

Un Sistema Operativo (SO) es en sí mismo un programa de computadora. Sin embargo, es un programa muy especial, quizá el más complejo e importante en una computadora. El SO despierta a la computadora y hace que reconozca a la CPU (Central Processing Unit), la memoria, el teclado, el sistema de vídeo, las unidades de disco, los puertos y conectores (serie y paralelo) de entrada y salida, además, proporciona la facilidad para que los usuarios se comuniquen con la computadora y sirve de plataforma a partir de la cual se corren programas de aplicación.

Los sistemas Operativos de Red empleados normalmente son UNIX, Linux, Macintosh OS, OS/2 y Windows 2000 Server. El UNIX y sus clones permiten múltiples tareas y múltiples usuarios. Otros Sistemas Operativos multiusuario y multitarea son OS/2, desarrollado inicialmente por Microsoft e IBM, Windows NT y Windows 2000 Server desarrollado por Microsoft. El sistema Operativo multitarea de Apple se denomina Macintosh OS.

Algunos de los sistemas operativos con los que cuenta Radio UNAM para el desarrollo del sistema de la fonoteca son:

Windows 98

Windows 98 una de las versiones del sistema operativo de Microsoft. Construido sobre las herramientas de vanguardia de Windows 95.

Características:

- Reducción del número de pasos necesarios para la instalación del equipo.
- Tiene más espacio en el disco, almacena información más eficientemente con la nueva herramienta FAT32.
- El asistente de mantenimiento programa y realiza automáticamente revisiones internas para que la computadora siempre esté funcionando en las mejores condiciones.

Los requisitos del sistema son:

- Procesador 486DX / 66 MHz o superior.
- 16 MB de memoria RAM.
- Una instalación típica requiere aproximadamente 195 Mb de espacio libre en el disco duro.

Ventajas:

- Necesita únicamente de una sola partición de 2 Gigas.
- El tamaño del "cluster" -unidad de asignación- es menor, por lo que los archivos ocupan menos espacio físicamente en el disco y por lo tanto es posible aprovechar mejor el disco. Típicamente en un disco de 2 Gb llega a un ahorro de espacio del 33%

Desventajas:

- Lento en el arranque.
- Es incompatible con otros sistemas operativos. Otros sistemas no trabajan con FAT 32.

Windows 2000 Server

Windows 2000 Server, es un Sistema Operativo de 32 bits, que puede hacer completo uso de los procesadores de estas características. Esto le permite sacar ventaja de los

microprocesadores más avanzados, como los Intel 80486 y posteriores. Además de ser multitarea, está diseñado para tomar ventaja del multiproceso simétrico.

Características:

- Permite el control total en el acceso al sistema y a los archivos o subdirectorios que hay en el sistema.
- Permite que las DLL's se instalen en los directorios de sus aplicaciones especificadas y eviten que se eliminen las DLL's compartidas.
- Evita la lentitud de respuesta, la sobrecarga de los sistemas, la colisión entre tareas y las colas de trabajo pesadas.
- Permite la gestión de usuarios, departamentos y delegaciones en dominios y subdominios.
- Establecimiento de políticas de permisos y accesos.

Los requerimientos del sistema son:

- Procesador Pentium 700 MHz ó mayor.
- 256 MB de memoria RAM.
- 2 GB de espacio en disco duro.

Ventajas:

- Administración desde la línea de comandos.
- Organización de clústeres (compatibilidad con ocho nodos).
- Red LAN inalámbrica segura (802.1X).
- Servicios de administración de emergencia: Compatibilidad con servidor desatendido.

- Herramientas Windows de Gestión

Desventajas:

- Lento en el arranque.
- Los costos por licencia son elevados.
- Poca compatibilidad con programas antiguos MSDOS.
- Se requiere como mínimo 256 MB en memoria RAM para un buen funcionamiento.
- Necesita de constante actualización, instalación de parches.

2.7 Lenguajes de Programación

Un lenguaje de programación es cualquier lenguaje artificial que puede utilizarse para definir una secuencia de instrucciones para ser procesados por una computadora. Es complicado definir qué es y qué no es un lenguaje de programación. Se asume generalmente que la traducción de las instrucciones a un código que comprende la computadora debe ser completamente sistemática. Normalmente es la computadora la que realiza la traducción.

Una vez identificada una tarea, el programador debe traducirla o codificarla a una lista de instrucciones que la computadora entienda. Un programa informático para determinada tarea puede escribirse en varios lenguajes. Según la función, el programador puede optar por el lenguaje que implique el programa menos complicado. También es importante que el programador elija el lenguaje más flexible y más ampliamente compatible para el caso de que el programa tenga varias aplicaciones.

Los lenguajes de programación con los que cuenta Radio UNAM para el desarrollo del sistema de la fonoteca son:

CAPÍTULO II

Visual Basic

Visual Basic se trata de un producto con una interfaz gráfica de usuario que sirve para crear aplicaciones para Windows basado en el lenguaje Basic y en la programación orientada a objetos.

La palabra “Visual” hace referencia al método que se utiliza para crear la interfaz gráfica del usuario. En lugar de escribir líneas de código para implementar una interfaz, se utiliza el ratón para arrastrar y colocar los objetos prefabricados al lugar deseado dentro de un formulario.

La palabra “Basic” hace referencia al lenguaje Basic (Beginners All-Purpose Symbolic Instruction Code). Visual Basic ha evolucionado a partir del lenguaje Basic original, y ahora contiene instrucciones, funciones y palabras clave, las cuales están directamente relacionadas con la interfaz gráfica de Windows.

Proporciona herramientas que permiten crear ventanas y controles sin escribir código. También incluye un entorno de desarrollo que permite ejecutar todas las tareas de edición, ejecución y mantenimiento de programas de una forma fácil y cómoda.

Visual Basic utiliza fundamentalmente dos herramientas, una que permite realizar los diseños gráficos y un lenguaje de alto nivel.

Visual Basic incluye como características las siguientes:

- Una biblioteca de clases que da soporte a los objetos Windows tales como ventanas, cajas de diálogo, controles.
- Un entorno de desarrollo integrado. Visual Basic fue diseñado para ser intérprete, lo que favorece la creación y la depuración de una aplicación.
- Galería de objetos incrustados y vinculados (OLE- Object Linking and Embedding).
- Visualización y manipulación de datos de otras aplicaciones Windows utilizando controles OLE.

- Una interfaz para múltiples documentos (MDI- Multiple Document Interface) que permite crear una aplicación con una ventana principal y múltiples ventanas de documento.
- Creación y utilización de bibliotecas dinámicas (DLL- Dynamic Link Libraries).
- Soporte para la programación de aplicaciones para Internet; forma parte de este soporte la tecnología de componentes activos (ActiveX).
- Acceso a bases de datos a través del control de datos ADO, utilizando el motor de Access o controladores ODBC.
- Biblioteca para SQL que permite manipular bases de datos relacionales.
- Un administrador visual de datos para manipular bases de datos.

Los requerimientos mínimos se indican a continuación:

- Microprocesador Pentium a 90 MHz o superior
- Disco duro con espacio mínimo disponible de 80 MB para realizar una instalación completa de la edición estándar
- Tarjeta de video soportada por Windows
- 32 MB de memoria RAM o superior
- Sistema operativo Windows 95 o superior
- Microsoft Internet Explorer versión 4.01 o posterior

Ventajas y desventajas de Visual Basic 6.0

La ventaja principal de este lenguaje de programación es su sencillez para programar aplicaciones de cierta complejidad para Windows.

Actualmente existen infinidad de controles ActiveX, denominación con la que se conoce a este tipo de componentes. Visual Basic facilita un número importante de ellos que en la mayoría de las ocasiones serán suficientes para el desarrollo de las aplicaciones.

La principal desventaja de Visual Basic 6.0 es la necesidad de archivos adicionales además del ejecutable y cierta lentitud en el momento de la ejecución en comparación con otros lenguajes. Hoy en día este último factor es cada vez menos determinante debido a la velocidad y capacidad del Hardware de última generación.

PHP

PHP corresponde a las iniciales de “Personal Home Page” Pocesador de Hipertexto. Su sintaxis es similar a los lenguajes “C” y “Perl”, que se interpretan por un servidor web Apache y genera código HTML dinámico, lo que permite crear un programa que se pueda ejecutar en el servidor desde un programa visualizador de páginas web y dar respuesta en función de los datos que introduzca el usuario.

Este lenguaje dispone de funciones para realizar las operaciones habituales de los lenguajes de programación, usar archivos, tratamiento de cadenas de texto, etc. Una de las características más interesantes es la facilidad para consultar bases de datos y generar páginas en función de los resultados obtenidos, además se pueden definir clases y usar ciertas características de la programación orientada a objetos.

Ventajas:

Presenta múltiples ventajas necesarias para el desarrollo de todo tipo de aplicaciones.

- Interfaz. No es necesario que el usuario aprenda nuevas combinaciones de teclas para aprender a programar, ya que se ejecuta a través de una interfaz familiar al usuario.

- Acceso a red. El programa se ejecuta en un servidor, al cual se puede tener acceso desde cualquier punto de la red.

- Protección del código. EL código esta protegido tanto de la manipulación de los usuarios como de la presencia de virus.

- Facilidad de aprendizaje. Cualquier persona que sepa algún lenguaje de programación puede aprender los fundamentos de PHP en un tiempo breve.

Desventajas:

- Todo el trabajo lo realiza el servidor y no delega al cliente. Por tanto puede ser más ineficiente a medida que las solicitudes aumenten de número.
- La legibilidad del código puede verse afectada al mezclar sentencias HTML y php.
- La orientación a objetos es aún muy deficiente para aplicaciones grandes.

Visual C++

Visual C++ es un entorno integrado de desarrollo que permite la programación orientada a objetos (POO) conjuntamente con el sistema de desarrollo SDK (también denominado API) de Windows. Al ser un entorno integrado Visual C++ incluye, entre otras, las siguientes herramientas de desarrollo:

- Editor de texto
- Compilador/Enlazador
- Depurador
- Visor de datos y dependencias (Browser)

La parte central del paquete de Visual C++ es Developer Studio (Estudio del Desarrollador), que es el Ambiente Integrado de Desarrollo (IDE, para los anglosajones).

Algunas de las herramientas que incorpora el Developer Studio son:

- Editor integrado, con capacidad de arrastrar y soltar, y resaltado en la sintaxis.
- Editor de recursos, para crear recursos de Windows, como mapas de bits, iconos, cuadros de diálogo y menús.

- Depurador integrado, permitiendo que se ejecuten programas y se revisen los posibles errores de forma muy sencilla.

- Ayuda en línea, sensible al contexto para todas las herramientas incluidas en el Developer Studio, incluyendo ayuda sobre el lenguaje C++, la interfaz de programación Windows y la biblioteca de clases MFC.

Características:

- Microsoft Windows 95/98 o Windows NT.

- Microsoft Visual C++

- Intel 486 o superior

- l6MB de RAM.

- Pantalla con resolución 800 x 600 o superior

- 20 MB de espacio de disco fijo libre.

- InstallWizard está diseñado para computadoras típicas de desarrollo, lo que significa un procesador Pentium y una resolución de 800 x 600 o superior. Se puede ejecutar lnstallWizard en una pantalla de 640 x 480, pero es posible que se necesite disponer de ventanas diferentes para ver toda la información.

A continuación se enlistan las ventajas de Visual C++.

- Presenta la posibilidad de accedeer a los Api's de Windows

- Posee soporte para OpenGL y Glide (son librerías de C).

- Posibilidad de interactuar con DirectX

- Potencia y compatibilidad

El siguiente listado muestra las desventajas de Visual C++.

- Toda la programación es íntegramente en código
- No es una entorno de ventanas y botones como lo es Delphi, Visual Basic o Borlan C++ Builder
- Consume demasiados recursos del sistema, el tiempo de compilación es relativamente alto.
- Precio bastante alto.

PROPUESTA DE SOLUCIÓN

3 Propuesta de Solución

La insuficiencia de recursos económicos, la escasa preparación del personal y la ausencia de voluntad política que inciden en la actual situación de las fonotecas de México que en palabras del Investigador Fernando Osorio (Radio Educación), requieren de una atención urgente, visto esto en el Segundo Seminario Nacional de Archivos Sonoros y Audiovisuales.

De acuerdo a una encuesta nacional llevada a cabo por Radio Educación para conocer el estado de las fonotecas en México se comprobó que en la mayoría de las fonotecas no existen condiciones adecuadas para la catalogación y preservación de los documentos sonoros.

Una fonoteca por definición es el lugar destinado a almacenar cintas ó alambres magnetofónicos, discos y todo tipo de grabaciones sonoras, en México existen tres tipos de fonotecas, las de Radiodifusión, las de Investigación y las de Apoyo Académico, cada una de estas fonotecas tienen sus objetivos y su problemática. Radio UNAM como ya se menciono cuenta con la fonoteca “Alejandro Gómez Arias” perteneciente a las fonotecas de Radiodifusión.

3.1 Requerimientos de Fonoteca

La fonoteca “Alejandro Gómez Arias” de Radio UNAM, tiene como principal objetivo el de resguardar y ordenar los más 120,000 soportes (cintas, CD, casetes) en las que se encuentra el testimonio cultural de la universidad y de algunos episodios importantes que forman parte de la historia del México contemporáneo. Tiene como funciones realizar el manejo rápido y adecuado de toda la información que se tiene, clasificándola de acuerdo a las políticas de la dirección de Radio UNAM, además de la difusión cultural a todos los interesados, es importante hacer notar que esta información es estructurada mediante series y programas, entendiendo como serie un conjunto de programas que se refieren al contexto de la serie. Para realizar estas funciones se realizan tres procesos:

1. Ingreso de soporte a la fonoteca (programa o serie)
2. Clasificación y almacenamiento del soporte.
3. Préstamo de soporte.

El proceso de ingreso de soporte a la fonoteca (Figura 3.1.1) consta de los siguientes puntos:

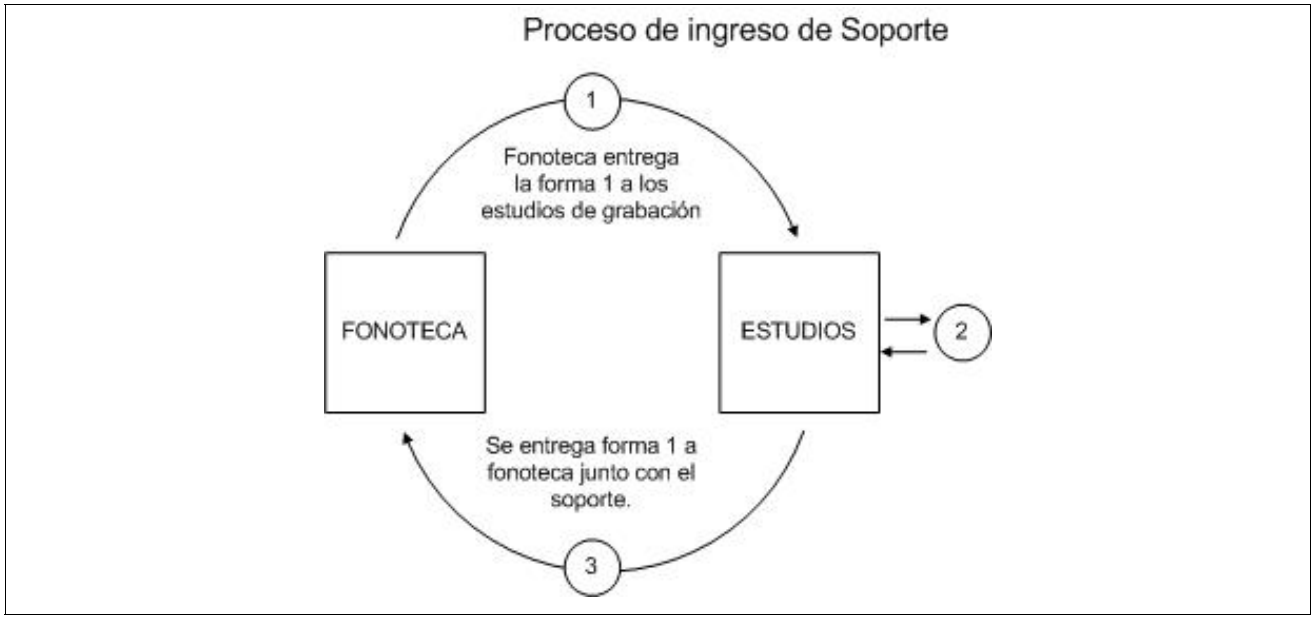
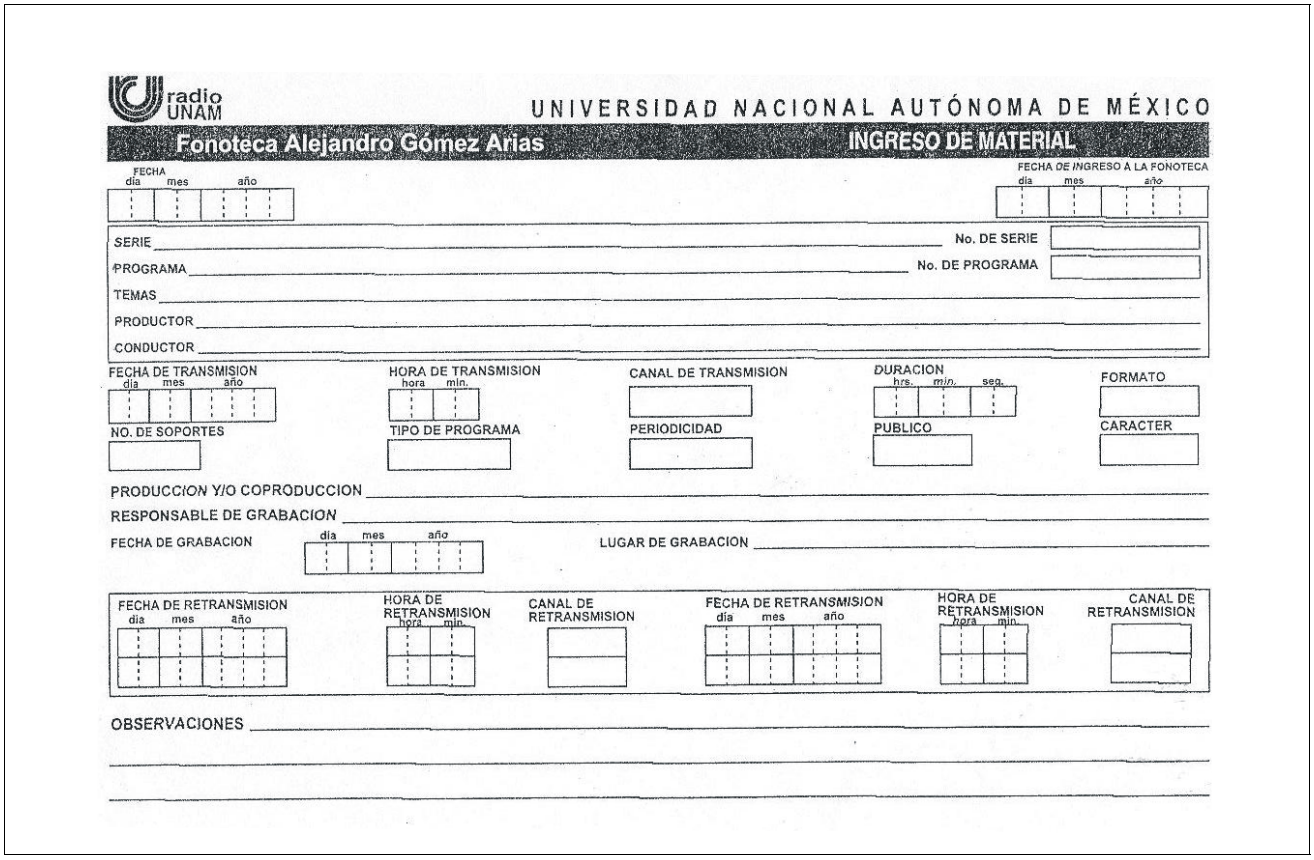


Figura 3.1.1 Procese de Ingreso de Soporte

- El departamento de fonoteca proporciona la forma de ingreso de material (figura 3.1.2) al departamento de producción, quien los distribuye a los productores cada vez que se realiza un programa.
- Al terminar el programa el productor llena la forma de ingreso de material (figura 3.1.2) proporcionando los campos que le son requeridos en fonoteca.
- El productor lleva el soporte y la forma de ingreso a la fonoteca en la cual se realiza el proceso 2.


radio UNAM

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Fonoteca Alejandro Gómez Arias — INGRESO DE MATERIAL

FECHA día mes año

FECHA DE INGRESO A LA FONOTECA día mes año

SERIE _____ No. DE SERIE

PROGRAMA _____ No. DE PROGRAMA

TEMAS _____

PRODUCTOR _____

CONDUCTOR _____

FECHA DE TRANSMISION día mes año

HORA DE TRANSMISION hora min.

CANAL DE TRANSMISION

DURACION hrs. min. seg.

FORMATO

NO. DE SOPORTES

TIPO DE PROGRAMA

PERIODICIDAD

PUBLICO

CARACTER

PRODUCCION Y/O COPRODUCCION _____

RESPONSABLE DE GRABACION _____

FECHA DE GRABACION día mes año

LUGAR DE GRABACION _____

FECHA DE RETRANSMISION día mes año

HORA DE RETRANSMISION hora min.

CANAL DE RETRANSMISION

FECHA DE RETRANSMISION día mes año

HORA DE RETRANSMISION hora min.

CANAL DE RETRANSMISION

OBSERVACIONES _____

Figura 3.1.2 Forma de Ingreso (Forma 1)

Los productores solo ocupan los campos de serie, programa, temas, productor, conductor y la fecha en que realizo el programa[1]

El proceso de clasificación y almacenamiento del soporte (figura 3.1.3) se lleva a cabo mediante los siguientes pasos:

[1] Las especificaciones de cada uno de los campos de la forma se muestran en el Diccionario de Datos.

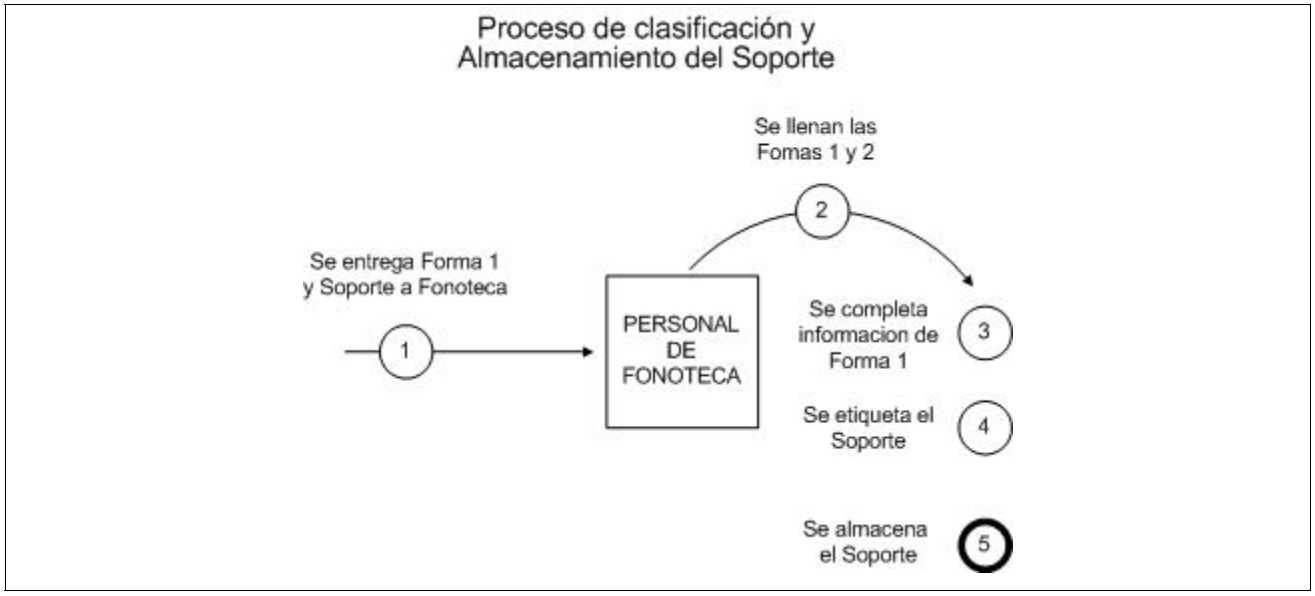


Figura 3.1.3 Proceso de Clasificación

- El soporte es entregado junto con la forma ingreso al personal de fonoteca.
- Se verifica si el soporte pertenece a una serie existente. Si la serie existe, únicamente es llenada la forma de programas (figura 3.1.4), en caso contrario se llenara la forma de series (figura 3.1.5) y el formato de programa (figura 3.1.4).
- El personal de fonoteca se encarga de completar la forma de ingreso de material clasificándolo por un número único de acuerdo a las políticas de este departamento.
- Los datos de la forma de ingreso son vaciados a una etiqueta que se adhiere al soporte, y se llena la ficha fonotecaría.
- Finalmente el soporte es almacenado en la bóveda de acuerdo a su clasificación.


radio UNAM

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Fonoteca Alejandro Gómez Arias — PROGRAMA

$035 FOLIO
$037 FECHA (día mes año)
$050 CLASIFICACIÓN
$020 LOCALIZACIÓN
$916 NUMERO DE CINTA
$930 FECHA DE TRANSMISIÓN (día mes año)
$931 HORA DE TRANSMISIÓN (hora min)
$084 DURACIÓN (hr min)
$932 CANAL DE TRANSMISIÓN
$929 FECHA DE GRABACIÓN (día mes año)
$008f PERIODICIDAD
$085 PÚBLICO
$914 TIPO DE PROGRAMA
$655 CARACTER
$076 FORMATO
$007 TIPO DE GRABACIÓN
$938 STATUS
$937 ESTADO FÍSICO DEL MASTER
$917 SOPORTES DEL PROGRAMA
$918 ADQUISICIÓN
$008a IDIOMA
$008&e PAÍS
$260a CIUDAD, ESTADO

$933 FECHA DE RETRANSMISIÓN (día mes año) | $934 HORA DE RETRANSMISIÓN (hora min) | $935 CANAL DE RETRANSMISIÓN
$933 FECHA DE RETRANSMISIÓN (día mes año) | $934 HORA DE RETRANSMISIÓN (hora min) | $935 CANAL DE RETRANSMISIÓN

$440a SERIE — $440q No. DE SERIE
$245 PROGRAMA — $440n No. DEL PROGRAMA EN LA SERIE
$650 TEMAS
$100 PRODUCTOR
$700 CONDUCTOR
$700 INVITADOS
$110 PRODUCCIÓN Y/O COPRODUCCIÓN
$700 RESPONSABLE DE GRABACIÓN
$911 LUGAR DE GRABACIÓN

$520 ANEXO — RESUMEN ◯ GUIÓN ◯ BITÁCORA O CONTENIDO ◯ OTRO, ESPECIFIQUE

$700 PARTICIPANTES DE PRODUCCIÓN

ASISTENTE DE PRODUCCIÓN
REALIZADOR
DIRECTOR
MUSICALIZADOR
EFECTOS
OPERACIÓN TÉCNICA
POST-PRODUCCIÓN
HISTORIA O IDEA
ORIGINAL
ADAPTACIÓN
GUIÓN
INVESTIGACIÓN
REPARTO
MÚSICA
VOCES
OTROS, ESPECIFIQUE

$500 OBSERVACIONES

$018 RESPONSABLES — CODIFICACIÓN — $016 FECHA DE CODIFICACIÓN (día mes año) — CAPTURA — $015 FECHA DE ALTA (día mes año) — MODIFICACIÓN — $017 FECHA DE MODIFICACIÓN (día mes año)

Figura 3.1.4 Forma de Programas (Forma 2)

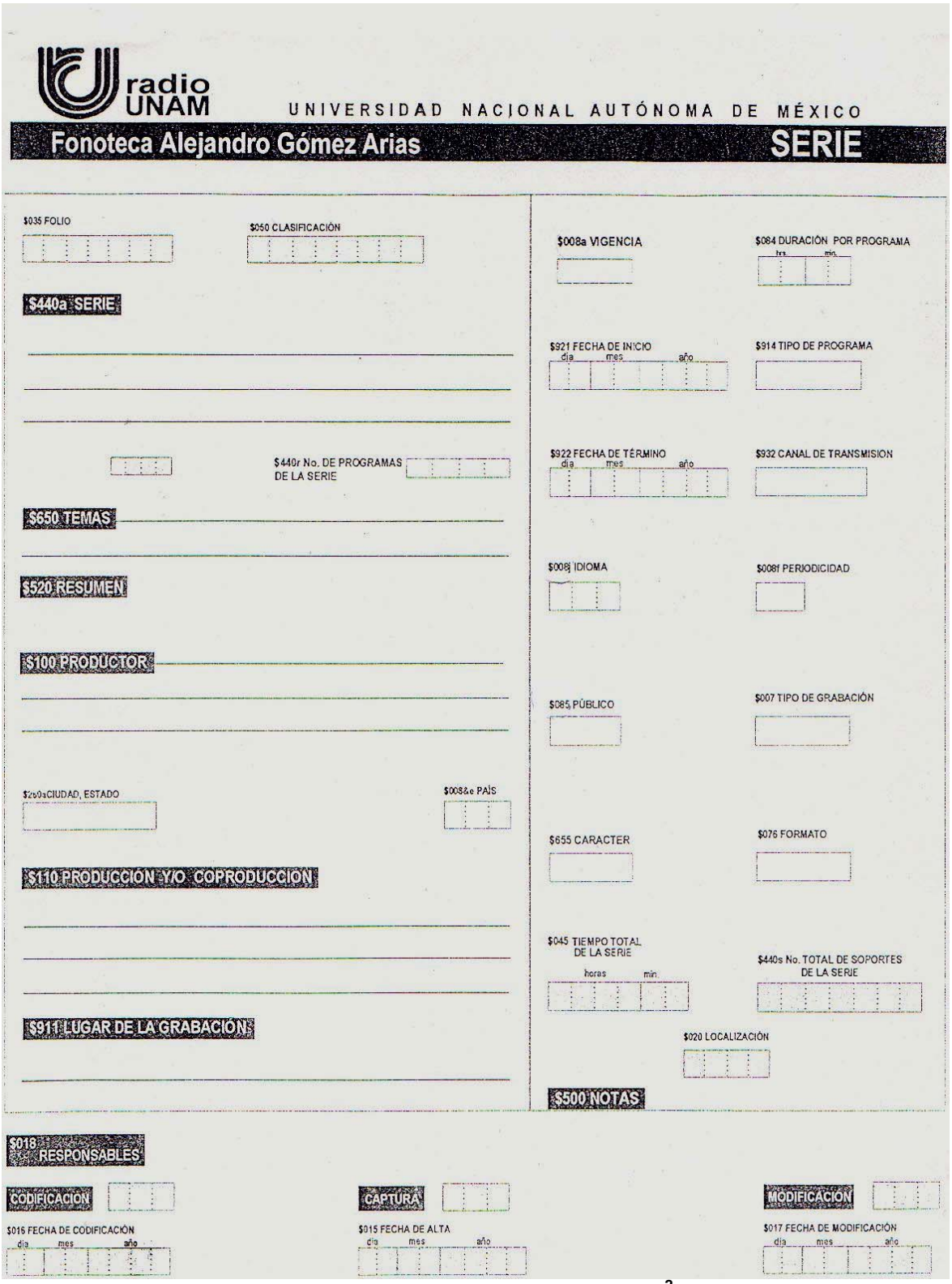


Figura 3.1.5 Forma de Serie (Forma 3)[2]

[2] Las especificaciones de los campos de las formas 2 y 3 se muestran en el Diccionario de Datos.

El proceso de Préstamo de Soporte (Figura 3.1.6) cuenta con los puntos:

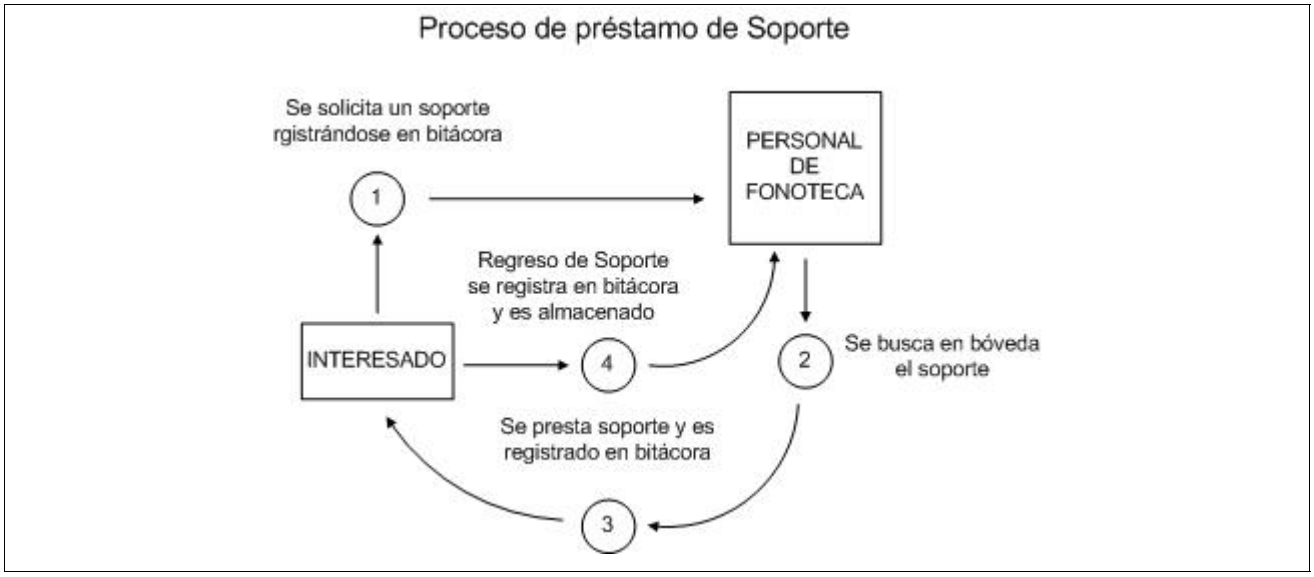


Figura 3.1.6 Proceso de Préstamo de Soporte

- Llega la persona solicitante de un soporte, proporcionando información sobre el, registrándose en la bitácora.

- El personal de este departamento se encarga de localizar la información proporcionada.

- Localizado el soporte, este es registrado en la bitácora correspondiendo con el interesado, a quien se le proporciona el material, indicando los cuidados y la fecha en que debe regresarlo.

- Cuando el material es devuelto, se registra en la bitácora y es almacenado nuevamente en la bóveda.

Cabe mencionar que el préstamo es únicamente interno, lo que quiere decir que el material no sale de las instalaciones de Radio UNAM.

La fonoteca busca tener un sistema computacional de manejo de base de datos que le permita cumplir con los puntos mencionados en el Capítulo I referentes a la problemática y necesidades de la fonoteca “Alejandro Gómez Arias”, además de realzar las tareas de búsqueda, inserción, actualización y obtención de reportes de manera automática, es decir con algoritmos

programados para estos fines.

3.2 Requerimientos del Sistema

El sistema computacional debe satisfacer con los siguientes requerimientos proporcionados por las necesidades de fonoteca.

Requerimientos generales

- El sistema deberá desarrollarse en una base de datos que pueda soportar la carga de trabajo en un ambiente multiusuario.
- La base de datos deberá soportar la cantidad de información contenida en el acervo.
- El sistema deberá mostrar diferentes formatos de visualización de la información según las necesidades de los usuarios (productores o encargados de Fonoteca).
- La posibilidad de imprimir reportes, principalmente información de series y programas, así como la actividad de préstamos.
- Deberá poseer un ambiente visual con el que este familiarizado el usuario final, es decir, de tipo Windows donde se muestran pantallas que contengan íconos y botones que muestren claramente las funciones que serán ejecutadas.
- Consistencia y confiabilidad en el manejo de la información.
- Que trabaje sobre las plataformas con las que cuenta la institución.
- Evitar en la manera de lo posible los errores humanos.

Requerimientos particulares.

- Generación y actualización de los catálogos que componen el sistema.
- Captura amigable de los datos, dado el nivel de experiencia de los usuarios.
- El borrado de cualquier tipo de información únicamente se realizara por personal autorizado, y directamente en el servidor de base de datos, no se tendrá esta opción en el sistema.
- Dependencia mínima del administrador del sistema y servicio de soporte técnico eficiente y rápido.
- Desarrollo de manual técnico y de usuario.
- Capacitación de los usuarios.

3.3 Infraestructura Computacional

Radio UNAM como ya se menciono es una dependencia de la Universidad Nacional Autónoma de México, y para lograr sus objetivos actualmente se apoya en IT (tecnología de la información), la cual se ha vuelto necesaria para esta dependencia, con esta IT se beneficia en aspectos relevantes, uno de ellos es la digitalización de sus cintas que hace posible un mayor almacenamiento de ellas en poco espacio (CD`s y DAT`s), además de que estas cintas digitalizadas se pueden compartir mediante la red LAN con que se cuenta, para diversos fines necesarios (edición y transmisión), también se comparten datos de interés en las diferentes áreas de la dependencia (documentos, correo y otros). Para poder tener estas y otras ventajas Radio UNAM cuenta con computadoras que soportan esta tecnología, cada una de ellas con el software adecuado para este fin, con esta IT se busca implementar el Sistema de Automatización de la fonoteca “Alejandro Gómez Arias” (SFAGA) para los objetivos de la misma. A continuación se describe la IT significativa para el objetivo de esta tesis.

La Red LAN corporativa con enlace de fibra óptica con Ciudad Universitaria, que es con la que se cuenta, se divide en dos subredes separadas físicamente una pública dividida en grupos de trabajo, y otra privada (Dalet), administradas por un solo servidor central. La conectividad de la red se realiza mediante dos Racks de comunicación localizados uno en el centro de cómputo y otro en la sala de grabación, enlazados por fibra óptica a siwtches 3com con velocidad de 10/100 Mbps organizados por apilamiento, a continuación se dan algunas definiciones necesarias en este punto.

En esta red LAN será implementado el SFAGA, en base a las características de las herramientas computacionales de IT con las que cuenta Radio UNAM, como son: sistemas operativos, manejadores de bases de datos y lenguajes de programación. A continuación dichas características se muestran.

Sistemas Operativos:

Windows 98, Windows 2000 Server.

Manejadores de Base de Datos.

SQL Server 2000, Microsoft ACCESS 2000 y MySQL

Lenguajes de Programación.

Visual Basic, PHP y Visual C++.

3.4 Opciones de Solución y Elección de la Óptima

Para la construcción de un sistema de información, la elección de herramientas es muy importante, para llevar a cabo dicha elección se tienen que tomar en cuenta aspectos tales como el tamaño de la aplicación, capacidad de los programadores, disponibilidad de equipo, tiempo de entrega del sistema y factores imprevistos, tomando en cuenta estos factores se deben analizar varias opciones de solución para posteriormente seleccionar la que mejor se ajuste a los requerimientos del sistema que se desea desarrollar.

En base a las necesidades vistas al principio de este capítulo requeridas por la fonoteca, la infraestructura IT con la que se cuenta y los factores antes mencionados se observan que el sistema a desarrollar debe cubrir los siguientes aspectos:

- La magnitud del sistema es considerado de tamaño medio, ya que el número de usuarios que tendrán acceso no es mayor a 10, la concurrencia entre usuarios es regular, la información contenida en el sistema es de tamaño considerable y el archivo histórico que se manejara es de 1 año.

- La plataforma de desarrollo es sistema operativo Windows montado sobre PC´s estándar.

- La plataforma sobre la que se montará el sistema es igualmente de computadores personales con sistema operativo Windows.

- La posibilidad de cambios en los requerimientos durante el desarrollo de la aplicación se estimó que será de un 70%, sin embargo también se ha evaluado que el impacto de estos cambios sobre la aplicación en general no será de consideración.

Dentro de los requerimientos de usuario se encuentra que el sistema debe manejar una interfaz de usuario intuitiva y sencilla basada en ventanas.

Teniendo en cuenta los puntos mencionados se generan varias opciones de solución con las herramientas gestoras de bases de datos y de programación mencionadas anteriormente.

La primera opción que se propondrá es la siguiente:

Herramienta de gestión de base de datos Access 2000 y herramienta de programación Visual Basic 6.0. esta opción presenta las siguientes ventajas:

- Access permite un manejo completo de SQL para hacer consultas y creación de objetos en la base en tiempo de ejecución.

- La integración entre estas dos herramientas es nativa y por lo tanto prácticamente transparente.

- Los formularios que se generen directamente en Access pueden ser reutilizados en Visual Basic sin ninguna modificación.

La principal desventaja de esta opción es que hay que tener cuidado con el uso de las herramientas de generación de código de Access y Visual Basic ya que generan un código demasiado pesado.

La segunda opción que se propondrá es la siguiente:

Herramienta de gestión de base de datos SQL Server 2000 y herramienta de programación Visual Basic 6.0. esta opción presenta las siguientes ventajas:

- Se cuenta con programadores con experiencia en programación en lenguaje Basic.
- SQL Server 2000 es una herramienta de gestión de base de datos de tamaño extra grande.
- SQL Server permite un mejor manejo de la seguridad que otras herramientas de su tamaño.
- SQL Server permite un manejo completo de SQL para hacer consultas y creación de objetos en la base en tiempo de ejecución.
- La integración entre estas dos herramientas es bastante completa.
- Visual Basic es una buena herramienta de desarrollo RAD (Rapid Aplication Development) para aplicaciones de tamaño mediano.
- El estilo de programación orientada a eventos se adapta bien al tipo de sistema que se pretende desarrollar.
- El soporte técnico y el número de código desarrollado sobre Visual Basic permiten una gran reutilización de componentes ya desarrollados que permiten la interacción con SQL Server.

- Visual Basic se puede enriquecer con herramientas de otras compañías que aumentan su poder y que son baratas y fáciles de implementar.

Como desventaja esta opción es que estas herramientas necesitan de actualizaciones constantes.

La tercera opción que se propondrá es la siguiente:

Herramienta de gestión de base de datos MySQL y herramienta de programación PHP presenta las siguientes ventajas:

- La integración PHP con MySQL genera óptimos resultados.
- No hay límites en el tamaño de los registros.
- Mejor control de acceso, en el sentido de qué usuarios tienen acceso a qué tablas y con qué permisos.

Se puede aprovechar de manera práctica que PHP viene acompañado por una excelente biblioteca de funciones que permite realizar cualquier labor (acceso a base de datos, encriptación, envió de correo, gestión de un e-commerce, xml, creación de PDF...) permitiendo una mejor conjunción con MySQL.

La desventaja que muestra este paquete es que para implementar esta opción es necesario tener instalado un servidor apache.

La cuarta opción que se propondrá es la siguiente:

Herramienta de gestión de base de datos SQL Server y herramienta de programación Visual C++ presenta la ventaja de poseer soporte para OpenGL y Glide (son librerías de C), que permiten la conexión con librerías predefinidas hacia ciertos manejadores de bases de datos como SQL Server

La desventaja que muestra este paquete es:

- La programación para relacionarlos es netamente en código.
- Los tiempos de compilación son muy largos.

En base al análisis anterior se optó por la segunda opción para desarrollar el Sistema “SAFAGA”, ya que cumple con:

- Las necesidades establecidas por parte de la fonoteca.
- Las herramientas se encuentran instaladas en los equipos principales de la fonoteca de Radio UNAM.
- Se conoce el ambiente de trabajo y desarrollo de ambas herramientas.
- Contienen las herramientas necesarias para la realización del Sistema.
- Son herramientas que están en constante actualización, por lo que sí se necesita migrar el sistema se puede realizar sin problemas, dado que las versiones son compatibles y mantiene la misma base en la que fue desarrollado el sistema. Además son herramientas que cuentan con constante soporte técnico y ayuda en línea por si se requiere.
- La integración de las herramientas es buena.

A los usuarios finales les ayudaría:

- La reducción de tiempos en cuanto a búsqueda de los soportes.
- Manejo con facilidad de la gran cantidad de información que conforma el acervo fonográfico.
- La información contenida en el sistema seria confiable y precisa.
- Una interacción amigable con el usuario.

DISEÑO, DESARROLLO E IMPLEMENTACIÓN DEL SISTEMA

4 Diseño, Desarrollo e Implementación del Sistema

4.1 Aplicación de la Metodología Elegida

Se eligió la metodología del análisis estructurado, en donde se divide un problema complejo en componentes más pequeños y se realizan relaciones definidas entre ellos. Esta metodología está principalmente orientada a procesos, concentrándose en las funciones del sistema requerido.

4.1.1 Diagrama de Contexto

Es también denominado modelo fundamental del sistema y lo representa como una sola burbuja o proceso, que identifica la función principal, con flujo de informaciones de entrada y salida, representadas por flechas, que lo relacionan con otros sistemas, personas u organizaciones (terminadores). El diagrama de contexto resume el requisito principal del sistema de recibir entradas, procesarlas de acuerdo con una determinada función y generar salidas. La figura 4.1.1. muestra el diagrama de contexto del sistema.

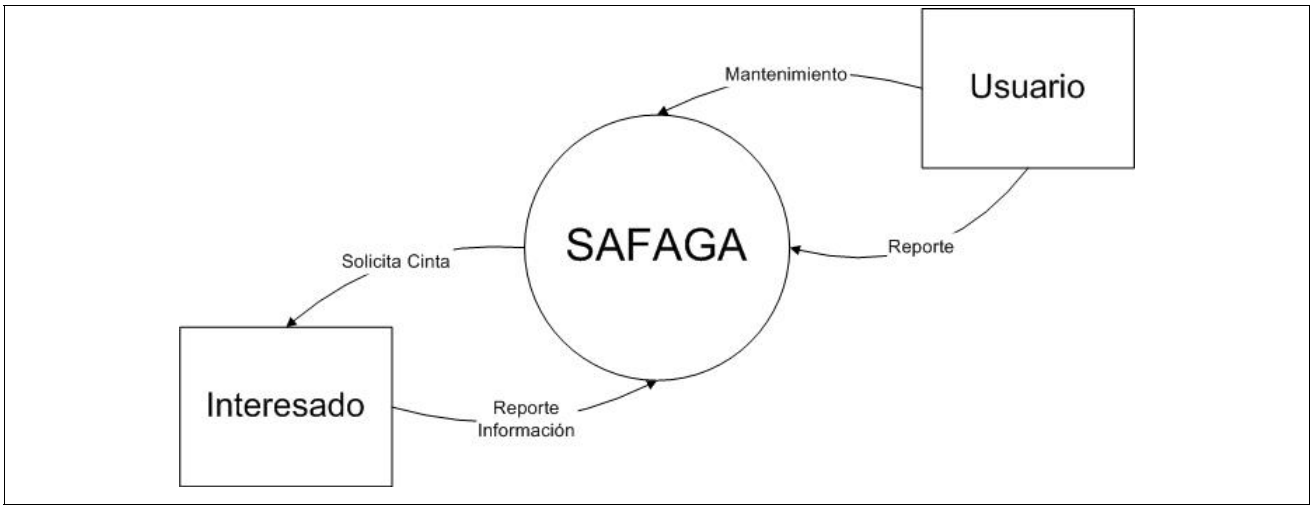


Figura 4.1.1 Diagrama de Contexto

4.1.2 Diagrama de Flujo de Datos

A partir del diagrama de contexto se construyó el diagrama de flujo de datos (DFD) como se muestra en la figura 4.1.2 en el que se define con mayor nivel de detalle los flujos de datos y procesos de transformación que ocurren en el sistema.

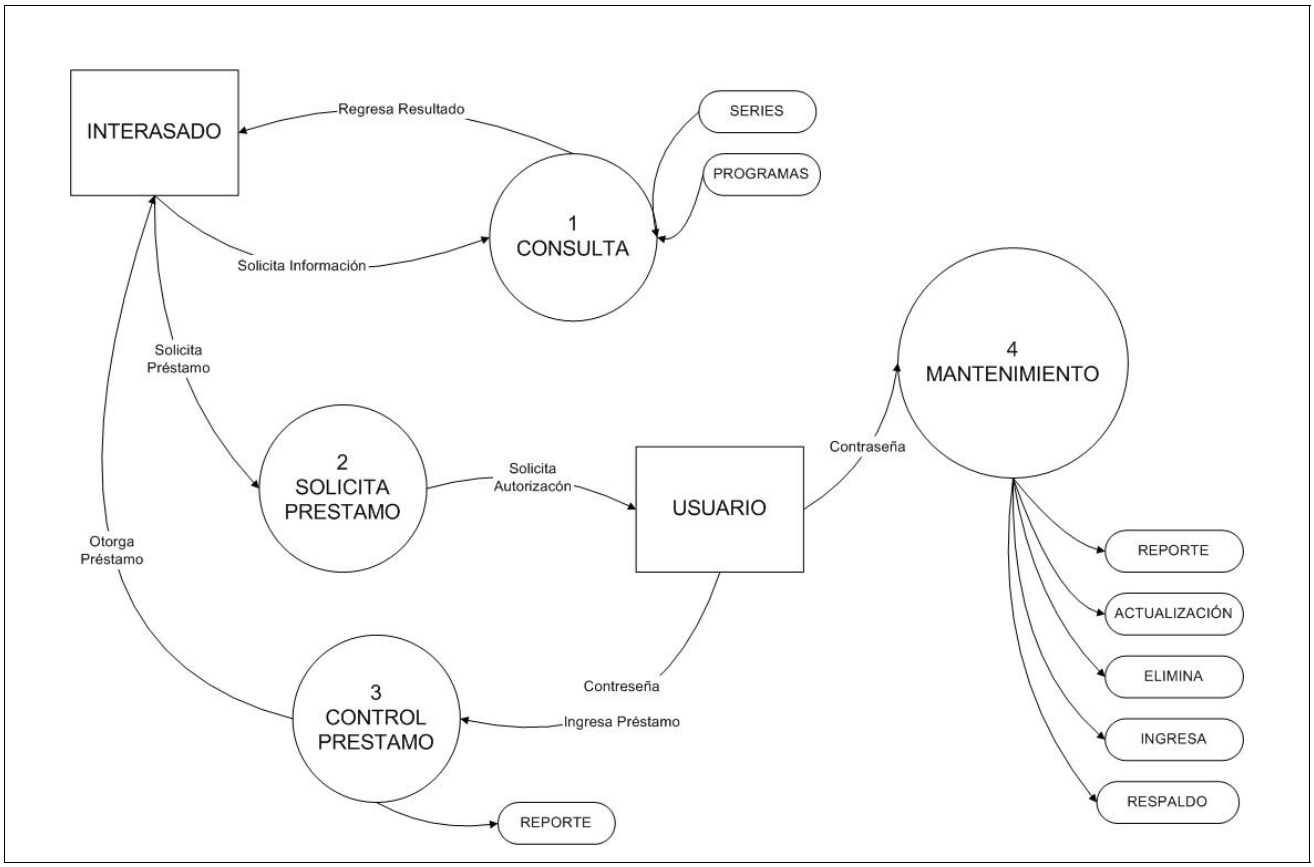


Figura 4.1.2 Diagrama de Flujo de Datos

Los procesos que se muestran en el DFD son los siguientes:

- Consulta. Este proceso permita tanto a usuarios como a interesados ver la lista completa de las series y los programas contenidos en el acervo de la fonoteca.

- Solicita Préstamo. En este proceso el interesado en alguno de los acervos se registra para tener un control de todos los préstamos de material realizados.

- Control Préstamo. En este proceso el encargado de la fonoteca da de alta a los interesados en el acervo, así como de llevar un informe del material que se encuentra en préstamo actualmente.

- Mantenimiento. El responsable de la fonoteca se encarga de introducir las fichas de las series y los programas nuevos, y de modificar las fichas existentes en caso de errores.

En la figura 4.1.3 se muestra que para realizar la consulta puede acceder tanto el usuario como el interesado, siendo el usuario el encargado de la fonoteca y el interesado la persona que solicita el préstamo del material, sin necesidad de una clave, pudiendo hacer la consulta por serie o por programa, en ambos casos se conecta a la base de datos (figura 4.1.4), realizando la búsqueda por los campos específicos, como nombre de la serie o programa, productor, conductor, entre otros.

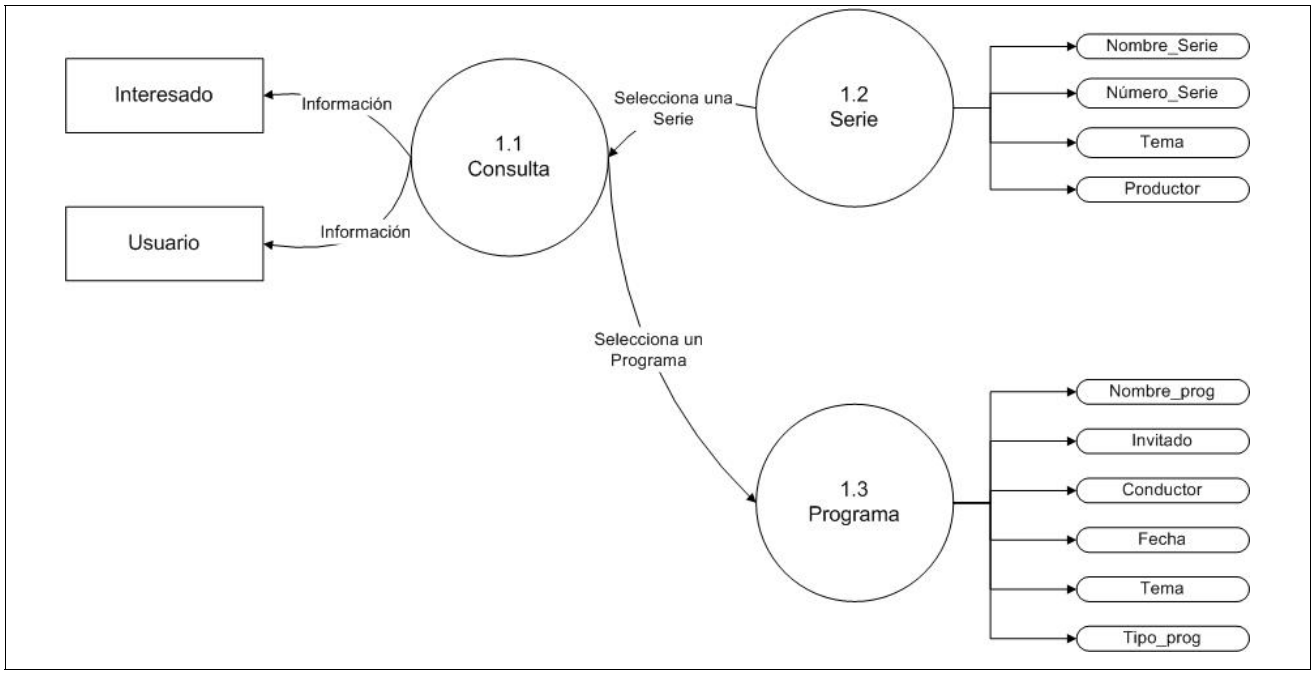


Figura 4.1.3 DFD 1: Consulta (Nivel 2)

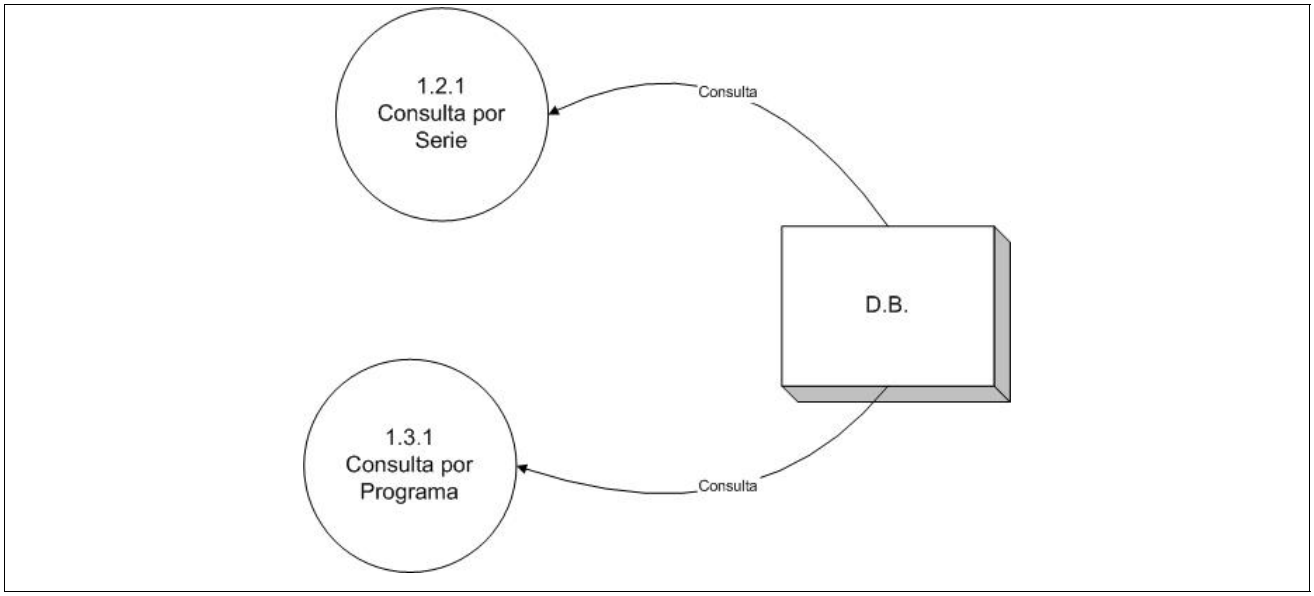


Figura 4.1.4 DFD 1.2: Consulta de Series o Programas (Nivel 3)

En la figura 4.1.5 el usuario necesita introducir una contraseña para saber si el interesado esta dado de alta en la base de datos, si no es así se le ingresa, si ya esta dado de alta se verifica si no tiene material en préstamo, si no es el caso se le otorga.

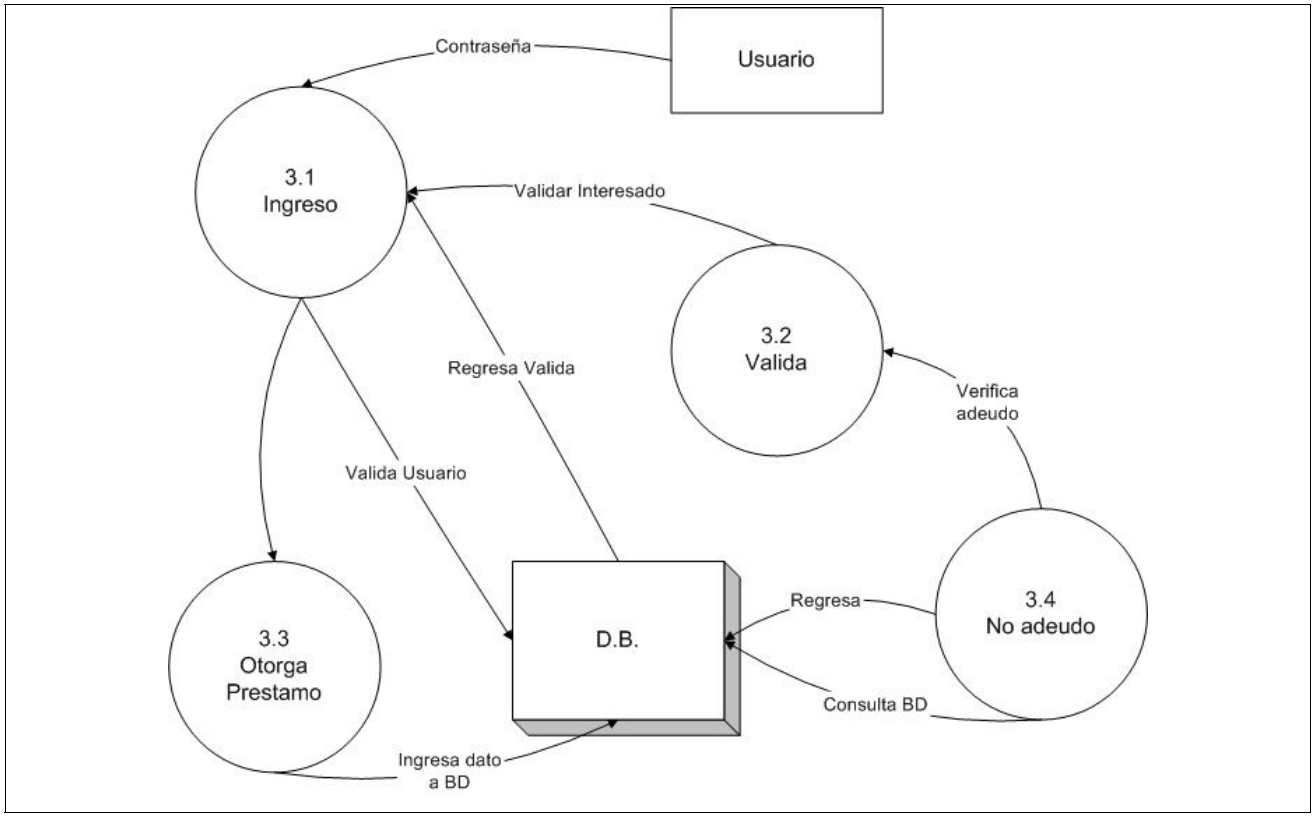


Figura 4.1.5 DFD 3.1: Control Préstamo (Nivel 2)

En la figura 4.1.6 se muestra como el usuario se valida antes de poder hacer modificaciones a los registros de las series y los programas, en caso de que no exista la ficha se crea, en caso contrario se actualizan los campos, generando un reporte de el usuario que realizó dicho proceso.

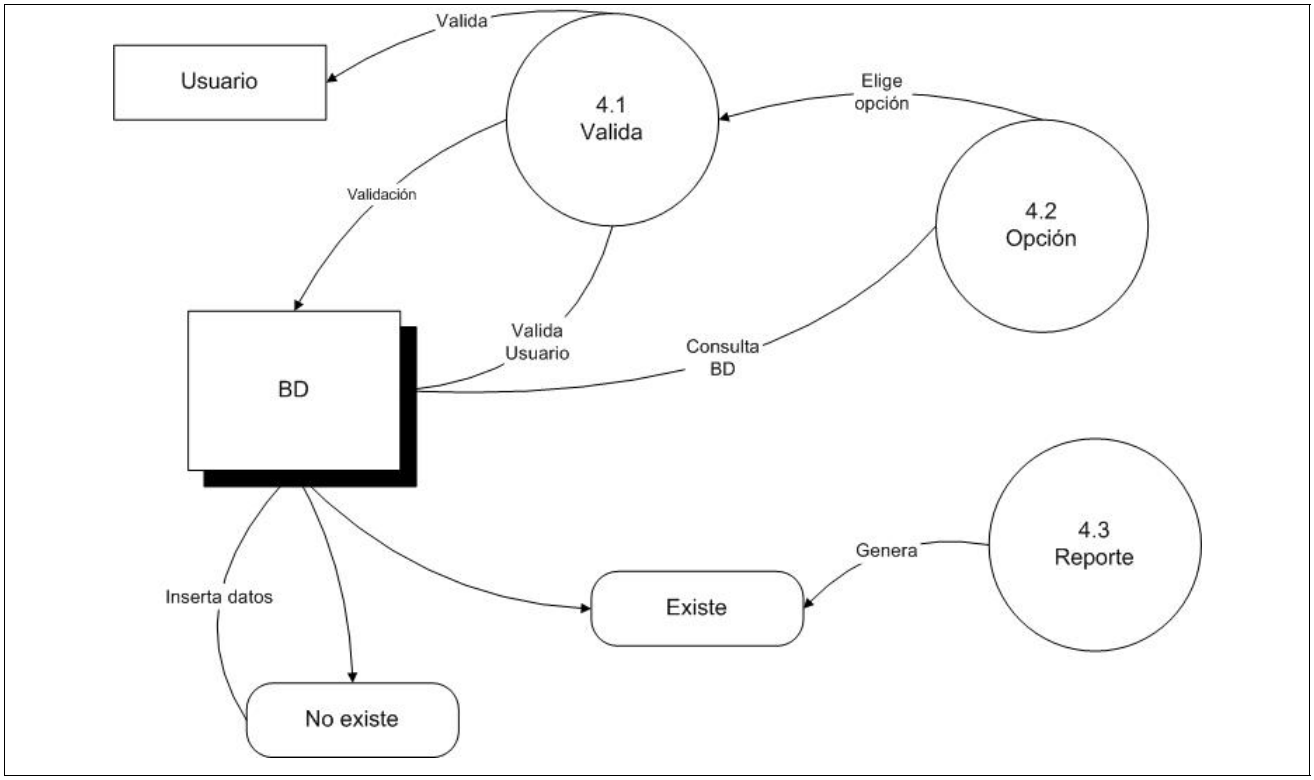


Figura 4.1.6 DFD 4.1: Mantenimiento (Nivel 2)

4.1.3 Diccionario de Datos

TABLA PROGRAMAS

Nombre de Atributo	Descripción	Tipo de Dato
Número de cinta (id_programa)	Código incremental para identificar cada ficha. No admite campos nulos. Llave primaria.	int
Número de serie (id_series)	Identifica a que serie pertenece el programa. Llave primaria de la tabla SERIES. Llave foránea.	int
Nombre del programa (nombre_p)	Nombre original del programa sin abreviaturas	varchar (512)
Conductor (conductor_p)	Nombre de la persona(s) que proporciona(n) la voz al programa.	varchar (120)
Fecha de codificación (fecha_codif_p)	Día, mes y año en que se codificó el programa	datetime (8)

Número de Programa de la Serie (num_prog_serie)	Número que identifica al programa dentro de una serie.	int
Fecha de captura (fecha_cap_p)	Indica día, mes y año en que se capturaron los datos del programa en la base de datos.	datetime (8)
Folio (folio_p)	Número de registros que se le otorgó a cada programa al ingresar a la fonoteca	varchar (15)
Duración (duración_p)	Indica las horas, minutos y segundos del tiempo de duración del programa.	varchar (18)
Fecha de grabación (fecha_grab_p)	Día, mes y año de grabación del programa	datetime (8)
Tipo programa (tipo_prog)	Indica el modelo característico del programa para su transmisión (vivo, grabado).	varchar (18)
Responsable de Grabación (res_grab_p)	Nombre del técnico encargado de la grabación del programa.	varchar (120)
Lugar de grabación (lugar_grab_p)	Indica el lugar físico en donde se grabo el programa	varchar (60)
Resumen (resumen_p)	Resumen del programa	varchar (1024)
Invitado (invitado_p)	Indica el nombre de los participantes en el programa, señalando la especialidad de cada uno	varchar (120)
Observaciones (obs_p)	Comentarios relacionados con el programa.	varchar (254)
Formato (id_formato)	Identifica a que formato pertenece. Funciona como llave foránea a la tabla FORMATO	int
Grabación (id_grabación)	Identifica a que tipo de grabación pertenece. Funciona como llave foránea a la tabla GRABACIÓN	int
Préstamo (id_prestamo)	Identifica el número de préstamo que se le asigno al material. Funciona como llave foránea a la tabla PRESTAMO	int
Transmisión (id_tp)	Identifica la ficha de los datos de transmisión del programa. Funciona como llave foránea a la tabla TRANSMISION	Int

TABLA SERIES

Nombre de Atributo	Descripción	Tipo de Dato
Número de serie (id_series)	Número que identifica a la serie. No admite valor nulo. Llave primaria	int
Clasificación (clasi_s)	Código de hasta 10 dígitos tomados de la clasificación del DEWEY de acuerdo al contenido temático de la serie	varchar (50)
Nombre de la serie (nombre_s)	Nombre original de la serie sin abreviaturas	varchar (512)
Folio de serie (folio_s)	Número de registros que se le otorgó a cada serie al ingresar a la fonoteca	varchar (15)
Número de programas (numprog_s)	Se indica el número total de programas que contiene la serie.	int
Tema_s (tema_s)	Tópico (s) que se aborde (n) en la serie	varchar (254)
Productor (productor_s)	Nombre de la persona responsable (intelectual) de la elaboración de la serie	varchar (120)
Ciudad (cuidad_s)	Nombre completo de la Ciudad y Estado donde se llevo a cabo la grabación de la serie	varchar (50)
País (id_pais)	Identifica en que país se realizo la serie. Funciona como llave foránea a la tabla PAIS	int
Producción y Coproduccción (proyco_s)	Nombre de la (s) institución (es) o dependencia (s) responsable (s) de la realización de la serie	varchar (120)
Fecha de inicio (fecha_ini_s)	Día, mes y año del primer programa de la serie al aire, emplear números arábigos	datetime (8)
Fecha de termino (fecha_fin_s)	Día, mes y año del último programa de serie al aire	datetime (8)
Idioma (id_idioma)	Identifica el idioma en que esta la serie. Funciona como llave foránea a la tabla IDIOMA	int
Tiempo total de la serie (tiempo_s)	Indica en horas y minutos la duración total de la serie	varchar (18)
Localización (localizacion_s)	Indica la ubicación de la serie dentro de bóveda	varchar (60)
Fecha de modificación (fecha_mod_s)	Día, ,mes y año en el cual se modifique o complete la información original	datetime (8)
Público (id_publico)	Identifica al tipo de publico al que va dirigido la serie. Funciona como llave foránea a la tabla PUBLICO	int
Perodo (id_periodo)	Identifica al periodo que corresponde a la serie. Funciona como llave foránea a la tabla PERIODO	int

Carácter (id_caracter)	Identifica el carácter de la serie. Funciona como llave foránea a la tabla CARACTER	int
Vigencia (vigencia_s)	Indica la posición de la serie en relación a su transmisión, al aire o fuera del aire.	varchar (15)
Fecha de codificación (fecha_codif_s)	Día, mes y año en que llego la serie a la fonoteca	datetime (8)
Transmisión (id_ts)	Identifica la ficha de los datos de transmisión de la serie. Funciona como llave foránea a la tabla TRANSMISION	int

Tabla Formato

Nombre de Atributo	Descripción	Tipo de Dato
Número Formato (id_formato)	Número incremental que identifica al formato. Funciona como llave primaria. No acepta valor nulo	int
Formato y/o soporte (tipo_formt_p)	Indica el tipo de contenedor en donde se grabo el programa: cinta, casete, DAT, CD, minidisc, otro.	varchar (25)

Tabla Grabacion

Nombre de Atributo	Descripción	Tipo de Dato
Número Grabación (id_grabacion)	Número incremental que identifica al tipo de grabación. Funciona como llave primaria. No acepta valor nulo	int
Grabación (tipo_grab_p)	Indica el tipo de grabación en que se realizo la serie, mono-aural, estéreo, digital, otro.	varchar (25)

Tabla Canaltryrtr

Nombre de Atributo	Descripción	Tipo de Dato
id_canaltrans	Número incremental que identifica el canal de transmisión y retransmisión. Funciona como llave primaria. No acepta valor nulo.	int
Canal de Transmisión (canal_trans)	Indica la frecuencia radiofónica por la cual se transmiten los programas, AM, FM, OC, Internet, otro.	varchar (25)

Tabla Público

Nombre de Atributo	Descripción	Tipo de Dato
id_publico	Número incremental que identifica al tipo de público. Funciona como llave primaria. No acepta valor nulo.	int
Público (tipo_pub_s)	Tipo de audiencia a la que se dirige la serie: general, adultos, adolescentes, infantil, otro.	varchar (25)

Tabla Carácter

Nombre de Atributo	Descripción	Tipo de Dato
Id_caracter	Número incremental que identifica el tipo de carácter. Funciona como llave primaria. No acepta valor nulo.	int
Carácter (tipo_car_s)	Objetivo de la serie: difusión, educativo, informativo, entretenimiento, otro.	varchar (25)

Tabla Periodo

Nombre de Atributo	Descripción	Tipo de Dato
id_periodo	Número incremental que identifica el tipo de periodo. Funciona como llave primaria. No acepta valor nulo.	int
Periodo (tipo_per_s)	Indica cada cuando se transmiten los programas de la serie.	varchar (25)

Tabla Responsable

Nombre de Atributo	Descripción	Tipo de Dato
id_respons	Número incremental que identifica a la persona responsable de capturar o modificar el programa o la serie. Funciona como llave primaria. No permite valores nulos.	int
Nombre del responsable (nombre_r)	Nombre completo de la persona responsable de la captura de la ficha	varchar (120)
Puesto del responsable (puesto_r)	Anotar el puesto de la persona responsable.	varchar (60)

Dirección del responsable (direccion_r)	Dirección de la persona responsable de la captura de la ficha, para su localización en caso de errores en la captura.	varchar (120)
Teléfono (telefono_r)	Teléfono del domicilio del responsable de capturar la ficha para su localización en caso de errores en la captura	varchar (20)
Fecha de alta del responsable. (fecha_alta_r)	Día, mes y año en que se le otorgó el permiso al responsable de la actualización de la base de datos.	datetime (8)
Clave del responsable (clave_r)	Palabra con la que el responsable se identifica para entrar al sistema.	varchar (10)
Contraseña del responsable (contraseña_r)	Palabra con la que se autentifica que el responsable sea el que dice ser.	varchar (8)

Tabla Interesado

Nombre de Atributo	Descripción	Tipo de Dato
Id_interesado	Número incremental que identifica al interesado de un préstamo. Funciona como llave primaria. No admite valores nulos.	int
Nombre del interesado (nombre_i)	Nombre completo de la persona interesada en un préstamo de material de la fonoteca.	varchar (120)
id_depto	Número que identifica al departamento al que pertenece el interesado. Funciona como llave foránea	int

Tabla Depto.

Nombre de Atributo	Descripción	Tipo de Dato
id_depto	Número incremental que identifica al departamento. Funciona como llave primaria. No admite valores nulos.	int
Nombre del departamento (nombre_d)	Nombre sin abreviaturas del departamento.	varchar (120)
Teléfono del departamento (tel_de)	Teléfono del departamento.	varchar (20)

Tabla Idioma.

Nombre de Atributo	Descripción	Tipo de Dato
Id_idioma	Número incremental que identifica al idioma. Funciona como llave primaria. No admite valores nulos.	int
Nombre del idioma (nombre_idioma)	Código de 3 caracteres tomados de la "revised list of lenguajes and lenguaje code 1997. Library of Congress" En cual indica el idioma o lenguaje original en el que se realizo la serie	varchar (25)

Tabla País.

Nombre de Atributo	Descripción	Tipo de Dato
Id_pais	Número incremental que identifica al país. Funciona como llave primaria. No admite valores nulos.	int
Nombre del país. (nombre_d)	Nombre sin abreviaturas del país.	varchar (25)
Siglas del País. (siglas_pais)	Código de 3 caracteres de acuerdo a la norma ISO 3166 [Code for the representation of names of contries]	varchar (5)

Tabla Préstamo.

Nombre de Atributo	Descripción	Tipo de Dato
id_prestamo	Número incremental que identifica el prestamo. Funciona como llave primaria. No admite valores nulos.	int
Fecha del préstamo (fecha_pres)	Fecha en el que fue prestado el material.	datetime (8)
Fecha de devolución (fecha_dev)	Fecha máxima en el que debería ser devuelto el material.	datetime (8)
Estado del préstamo (edo_pres)	Especifica si el préstamo esta vigente o ya paso la fecha en que debería ser devuelto.	int
Persona que recibió el material (res_cierre)	Contiene el nombre del responsable que recibió el material devuelto.	int
Fecha devolución real (fecha_cierre)	Fecha exacta en que el material fue devuelto.	datetime (8)
id_interesado	Número que identifica a la persona que quiere el material prestado. Funciona como llave foránea.	int

id_respons	Numero que identifica a la persona que otorgo el préstamo del material. Funciona como llave foránea.	int

Tabla TRANSMISION

Nombre de Atributo	Descripción	Tipo de Dato
Id_tp	Número incremental que identifica la fecha de transmisión y sus retransmisiones del programa o la serie. Funciona como llave primaria. No admite valores nulos.	int
Fecha de transmisión (ft)	Indica la fecha en que se transmitió el programa o la serie por primera vez.	datetime (8)
Hora de transmisión (ht)	Indica la hora en que se transmitió por primera vez el programa o la serie	varchar (15)
Canal de transmisión (ct)	Indica el canal por el cual se transmitió el programa o la serie la primera vez.	varchar (50)
Fecha de transmisión 1 (ft1)	Indica la fecha en que se retransmitió el programa o la serie por primera vez	datetime (8)
Hora de transmisión 1 (ht1)	Indica la hora en que se retransmitió por primera vez el programa o la serie.	varchar (15)
Canal de transmisión 1 (ct1)	Indica el canal por el cual se retransmitió el programa o la serie la primera vez.	varchar (50)
Fecha de transmisión 2 (ft2)	Indica la fecha en que se retransmitió el programa o la serie por segunda vez.	varchar (15)
Hora de transmisión 2 (ht2)	Indica la hora en que se retransmitió por segunda vez el programa o la serie.	varchar (15)
Canal de transmisión 2 (ct2)	Indica el canal por el cual se retransmitió el programa o la serie la segunda vez.	varchar (50)
Fecha de transmisión 3 (ft3)	Indica la fecha en que se retransmitió el programa o la serie por tercera vez.	datetime (8))
Hora de transmisión 3 (ht3)	Indica la hora en que se retransmitió por tercera vez el programa o la serie.	varchar (15)
Canal de transmisión 3 (ct3)	Indica el canal por el cual se retransmitió el programa o la serie la tercera vez.	varchar (50)
Fecha de transmisión 4 (ft4)	Indica la fecha en que se retransmitió el programa o la serie por cuarta vez.	datetime (8)
Hora de transmisión 4 (ht4)	Indica la hora en que se retransmitió por cuarta vez el programa o la serie.	varchar (15)
Canal de transmisión 4 (ct4)	Indica el canal por el cual se retransmitió el programa o la serie la cuarta vez.	varchar (50)
Fecha de transmisión 5 (ft5)	Indica la fecha en que se retransmitió el programa o la serie por quinta vez.	datetime (8)

Hora de transmisión 5 (ht5)	Indica la hora en que se retransmitió por quinta vez el programa o la serie.	varchar (15)
Canal de transmisión 5 (ct5)	Indica el canal por el cual se retransmitió el programa o la serie la quinta vez.	varchar (50)
Fecha de transmisión 6 (ft6)	Indica la fecha en que se retransmitió el programa o la serie por sexta vez.	datetime (8)
Hora de transmisión 6 (ht6)	Indica la hora en que se retransmitió por sexta vez el programa o la serie.	varchar (15)
Canal de transmisión 6 (ct6)	Indica el canal por el cual se retransmitió el programa o la serie la sexta vez.	varchar (50)
Fecha de transmisión 7 (ft7)	Indica la fecha en que se retransmitió el programa o la serie por séptima vez.	datetime (8)
Hora de transmisión 7 (ht7)	Indica la hora en que se retransmitió por séptima vez el programa o la serie.	varchar (15)
Canal de transmisión 7 (ct7)	Indica el canal por el cual se retransmitió el programa o la serie la séptima vez.	varchar (50)

4.1.4 Modelo Entidad-Relación

El diagrama entidad-relación representa las relaciones entre los objetos de datos, es por eso que se utiliza como herramienta para resaltar el ambiente en el cual interactúan los datos, así como la cardinalidad que existe entre cada uno de ellos.

El diagrama entidad-relación del sistema de automatización de la fonoteca “Alejandro Gómez Arias” se muestra en la figura 4.1.7

Basándose en el diagrama entidad-relación se obtiene el modelo entidad-relación, el cual lleva a obtener un modelo de datos relacional que no es otra cosa que una colección finita de tablas de dos dimensiones formadas por columnas y filas que representan una situación (véase capítulo II).

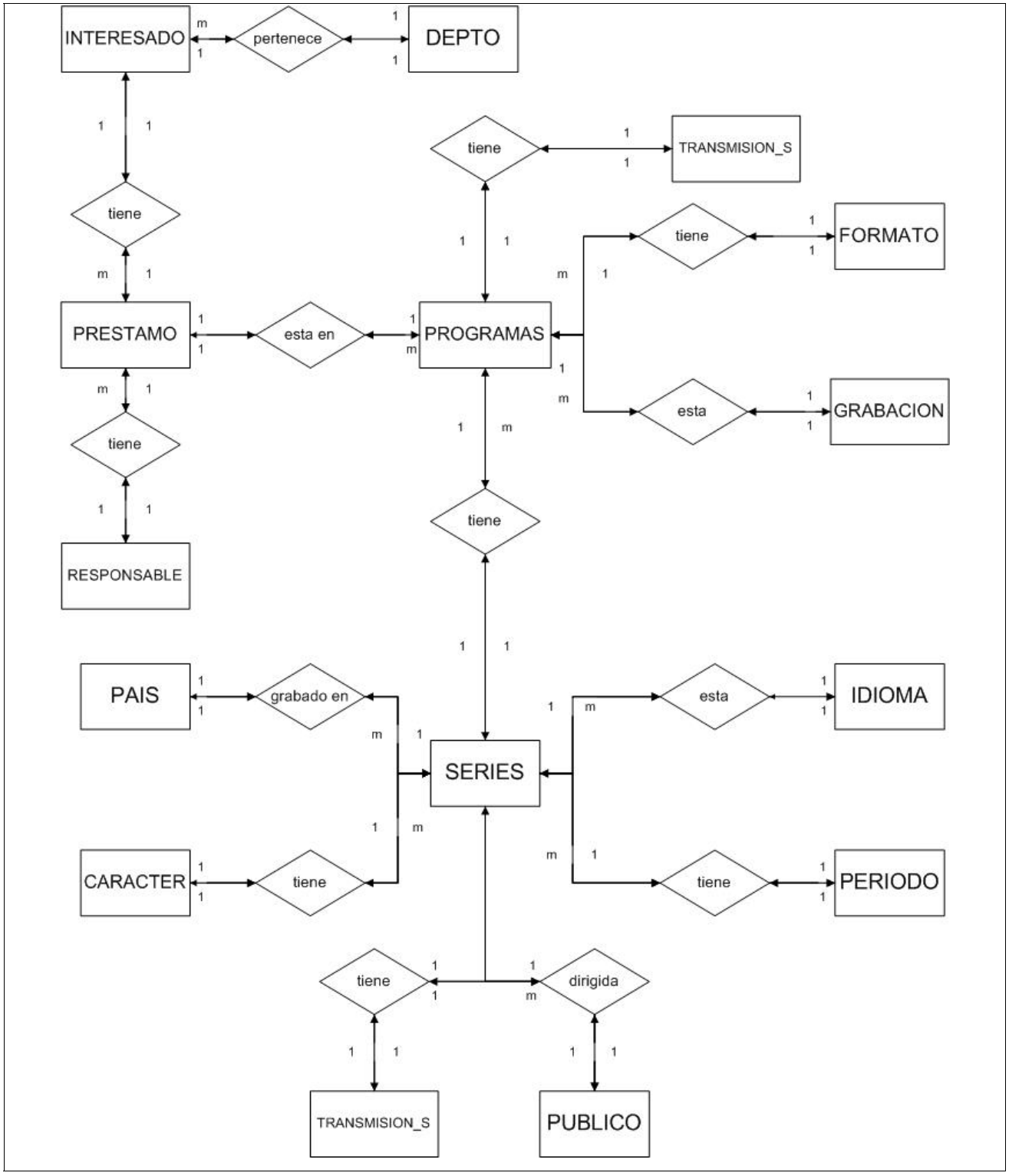


Figura 4.1.7 Diagrama Entidad-Relación

En la figura 4.1.8 se muestra el modelo entidad-relación con una tabla por cada entidad.

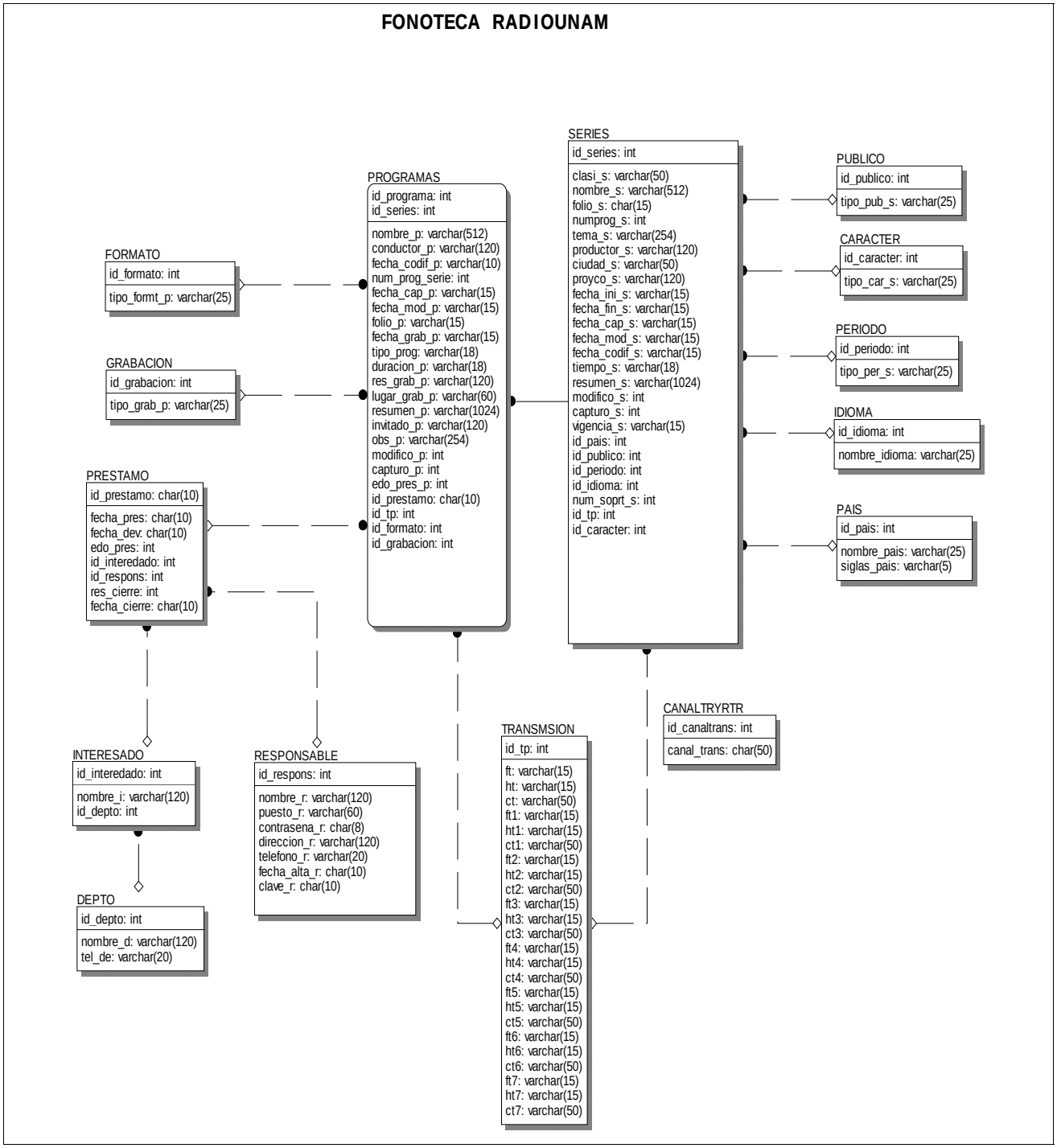


Figura 4.1.8 Modelo Entidad-Relación

CAPÍTULO IV

Para la definición de las entidades que forman la base del modelo desarrollado se consideraron las siguientes:

- Series: registra las series existentes en la fonoteca, con indicaciones de fecha de realización, nombre del productor, fecha de ingreso y otros datos generales.
- Programas: registra todos los programas existentes en la fonoteca, indicando a que serie pertenecen, fecha de realización, fecha de ingreso, entre otros datos generales.
- Público: incluye todos los tipos de público para los que esta dirigida la serie.
- Carácter: contiene todos los tipos de carácter que puede ser la serie.
- Periodo: contiene los tipos de periodicidad con que se transmite la serie.
- Idioma: incluye los posibles idiomas en los que puede estar realizada la serie.
- País: registro de los países en los que se pudo realizar la serie.
- Formato: contiene todos los tipos de formato en los que puede estar almacenado un programa.
- Grabación: registra los tipos probables en los que se pudo grabar el programa.
- Transmisión: contiene la fecha de transmisión del programa o la serie, así como sus retransmisiones, si existen.
- CanlTryRtr: incluye los canales de transmisión y retransmisión.
- Interesado: contiene el nombre de la persona a la que se le prestó el material, así como el departamento al que pertenece.
- Depto: registro de los departamentos de Radio UNAM, con la extensión telefónica correspondiente.

- Responsable: contiene el nombre de la persona encargada de dar de alta la serie o el programa, entre otros datos generales del usuario.

- Préstamo: registra la información de cuándo, quien y a quién se le prestó material de la fonoteca.

4.1.5 Normalización

El proceso de normalización como ya se vio en el capítulo II es un estándar que consiste en un proceso de conversión de las relaciones entre las entidades, evitando la redundancia de los datos, inconsistencia, pérdidas no intencionadas y la imposibilidad de introducir datos debido a la ausencia de otros datos.

Primer Forma Normal.

Se tiene una Tabla A (Tabla 4.1.1) que contiene la información de los programas y las series

id_serie	serie	programa
1	PANORAMA	JAZZ ACTUAL, JAZZ MEXICANO, JAZZ DE JUNIO
2	CANTO	CANTO NUEVO, CANTO AZUL
3	CONCIERTO	EDII GOMEZ I, EDII GOMEZ II
4	PERIODISMO	POLITICA DE HOY

Tabla 4.1.1 Tabla A

Se puede observar que la tabla A no cumple con la primer forma normal, ya que el atributo programa, no contiene un único valor. Es necesario crear una tabla adicional, obteniendo las siguientes tablas C y D.

id_serie	serie
1	PANORAMA
2	CANTO
3	CONCIERTO
4	PERIODISMO

Tabla 4.1.2 Tabla C en 1 FN

	id_programa	programa
	1	JAZZ ACTUAL
	2	JAZZ MEXICANO
	3	JAZZ DE JUNIO
	4	EDII GOMEZ I
	5	EDII GOMEZ II
	6	POLITICA DE HOY
	7	CANTO NUEVO
	8	CANO AZUL

Tabla 4.1.3 Tabla D en 1 FN

Segunda Forma Normal

Se tiene una tabla F, la cual contiene los programas pertenecientes a cada serie

	id_serie	id_programa	programa	serie
	1	1	JAZZ ACTUAL	PANORAMA
	2	8	CANTO AZUL	CANTO
	3	5	EDII I	CONCIERTO
	4	6	POLITICA DE HOY	PERIODISMO

Tabla 4.1.4 Tala F en 1 FN

Considerando que la anterior tabla tiene su llave primaria compuesta por id_serie y id_programa, se observa que todos sus atributos son únicos, lo que indica que se encuentra en primera forma normal, ahora se verifica para la segunda forma normal.

El atributo llamado programa depende funcionalmente de la llave compuesta (por el contexto de los datos), mientras que el atributo serie no cumple con lo anterior, únicamente depende de la llave id_serie, por lo que la tabla no se encuentra en segunda forma normal, por lo que se tiene:

	id_serie	serie
	1	PANORAMA
	2	CANTO
	3	CONCIERTO
	4	PERIODISMO

Tabla 4.1.5 Tabla G en 2 FN

id_serie	id_programa	programa
1	1	JAZZ ACTUAL
2	8	CANTO AZUL
3	5	EDII I
4	6	POLITICA DE HOY

Tabla 4.1.6 Tabla H en 2 FN

Con lo que se obtiene la segunda forma normal para cada una de las tablas.

Tercer Forma Normal

Se tiene una tabla M que contiene los datos de los interesados para obtener préstamos de programas grabados, y una tabla N que contiene el nombre de los departamentos a los que pertenecen dichos interesados.

id_interesado	interesado	direccion	tel_departamento	id_departamento
1	JOSE LUIS	AV.ZACATEACAS 21	753-896-90	1
2	SANTIAGO	CDA. AJUSCO 65	753-896-01	1
3	EDUARDO	AV. EUGENIA 190	753-896-65	2
4	JOAQUIN	AV.MEXICO 1235	753-896-98	3

Tabla 4.1.7 Tabla M en 2 FN

id_departamento	departamento
1	FONOTECA
2	DISCOTECA
3	DIRECCION

Tabla 4.1.8 Tabla N en 2 FN

Se verifica para la tercera forma normal en la Tabla M, se observa que los atributos dependen directamente del id_interesado, únicamente el atributo tel_departamento es quien tiene una mayor dependencia con la tabla N (id_departamento) que con la tabla M (id_interesado). Por lo que la tabla M no se encuentra en tercer forma normal.

Se verifica la tercer forma normal para la Tabla N, se observa que el atributo departamento tiene valores únicos y depende directamente del id_departamento, por lo que se encuentra en tercer forma normal.

Para que M se encuentre en tercera forma normar se realiza:

	id_interesado	interesado	direccion	id_departamento
	1	JOSE LUIS	AV.ZACATEACAS 21	1
	2	SANTIAGO	CDA. AJUSCO 65	1
	3	EDUARDO	AV. EUGENIA 190	2
	4	JOAQUIN	AV.MEXICO 1235	3

Tabla 4.1.9 Tabla M en 3 FN

	id_departamento	departamento	tel_departamento
	1	FONOTECA	753-896-90
	2	DISCOTECA	753-896-65
	3	DIRECCION	753-896-98

Tabla 4.1.10 Tabla N en 3 FN

Se observa ahora que las dos tablas están en tercera forma normal.

4.2 Diseño y Construcción del Back End

Como se mencionó anteriormente para la construcción de la base de datos (Back End) se utiliza SQL Server. En los siguientes renglones se describirá el método para la construcción de las tablas que componen la base de datos.

Al iniciar la aplicación se debe indicar sobre que base de datos se va a trabajar, en caso de que no exista se crea la base, como se muestra en la figura 4.2.1

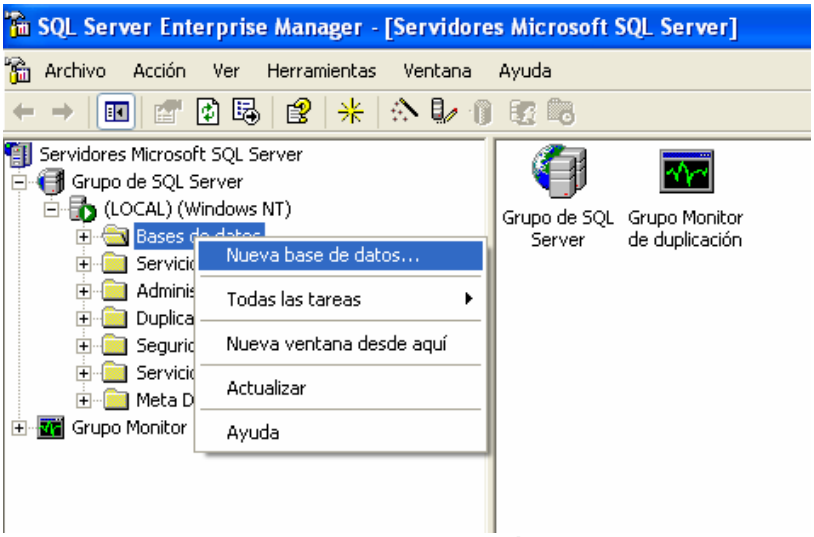


Figura 4.2.1 Pantalla de Creación SQL Server

A continuación se abre el analizador de consultas SQL para crear las tablas mediante código, ya sobre la base que se creó para la aplicación, como se muestra en la figura 4.2.2

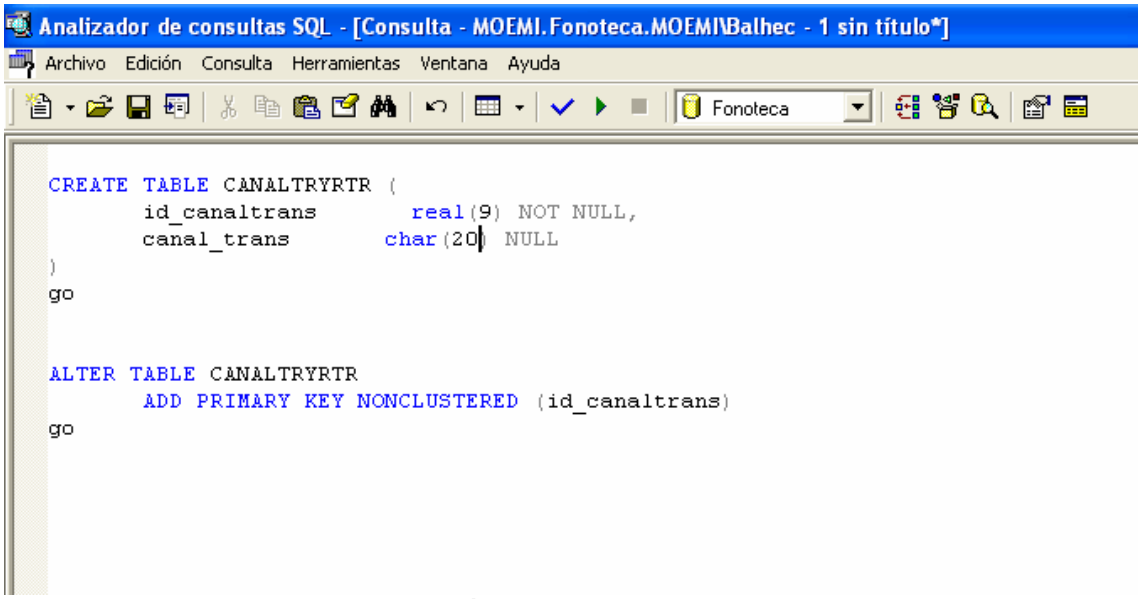


Figura 4.2.2 Creación de las tablas de la base de datos

Para simplificar esta tarea se utilizó el software Erwin, el cual permite crear el modelo de la base de datos relacional de una forma gráfica, pasando después todo a código sql el cual se puede interpretar desde SQL Server para la creación de la base de datos (ver apéndice C),

señalando cuales atributos funcionan como llaves primarias, cuales como llaves foráneas, así como el tipo y longitud del dato de cada atributo. En la figura 4.2.3 se muestra un ejemplo de la realización de una tabla.

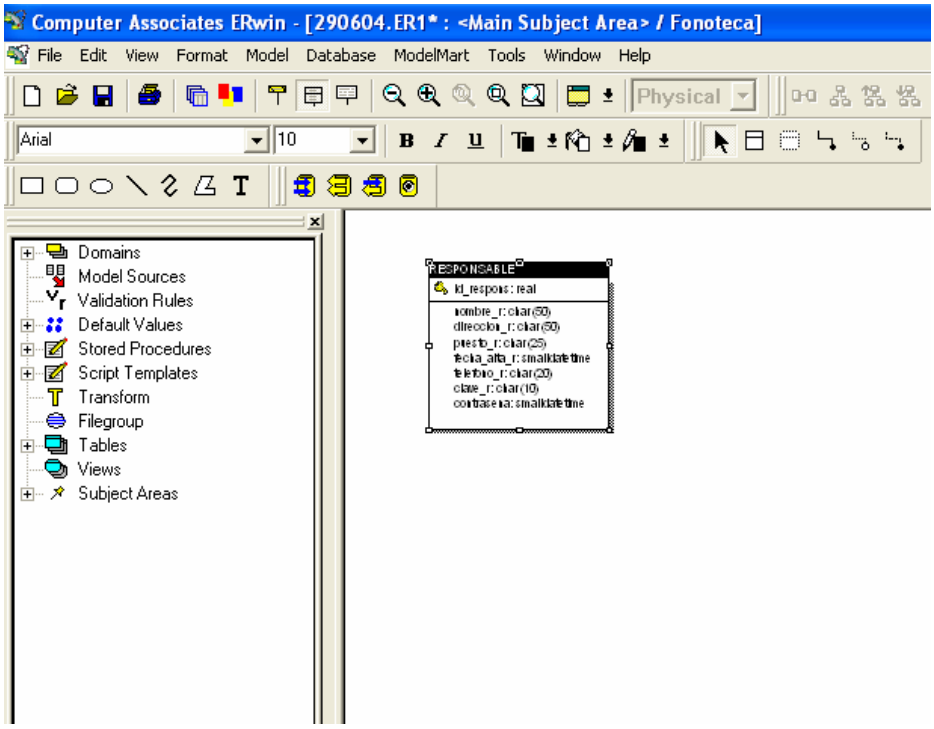


Figura 4.2.3 Creación de las tablas con ayuda de Edwin

Una vez que se realizaron todas las tablas con sus especificaciones y las relaciones que existen entre ellas se genera el código con una herramienta que contiene el software Erwin, la cual se muestra en la figura 4.2.4.

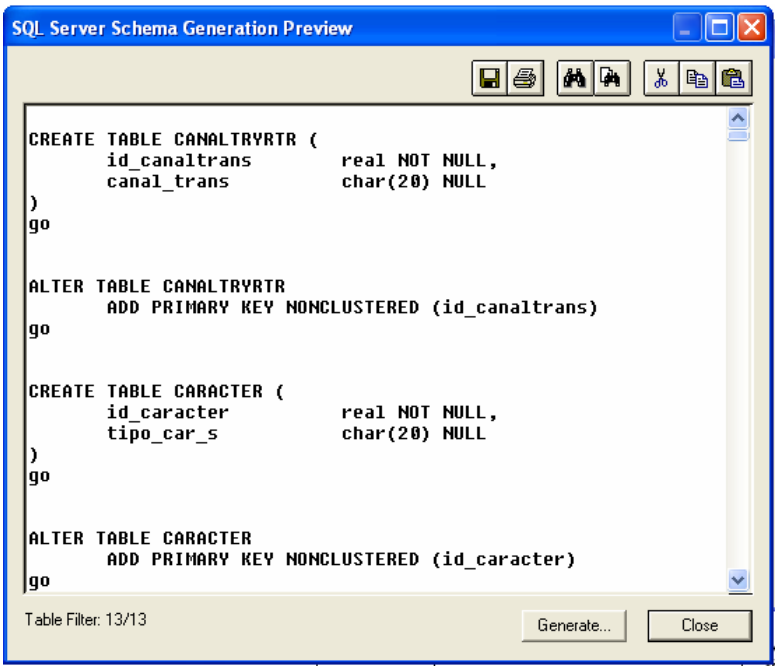
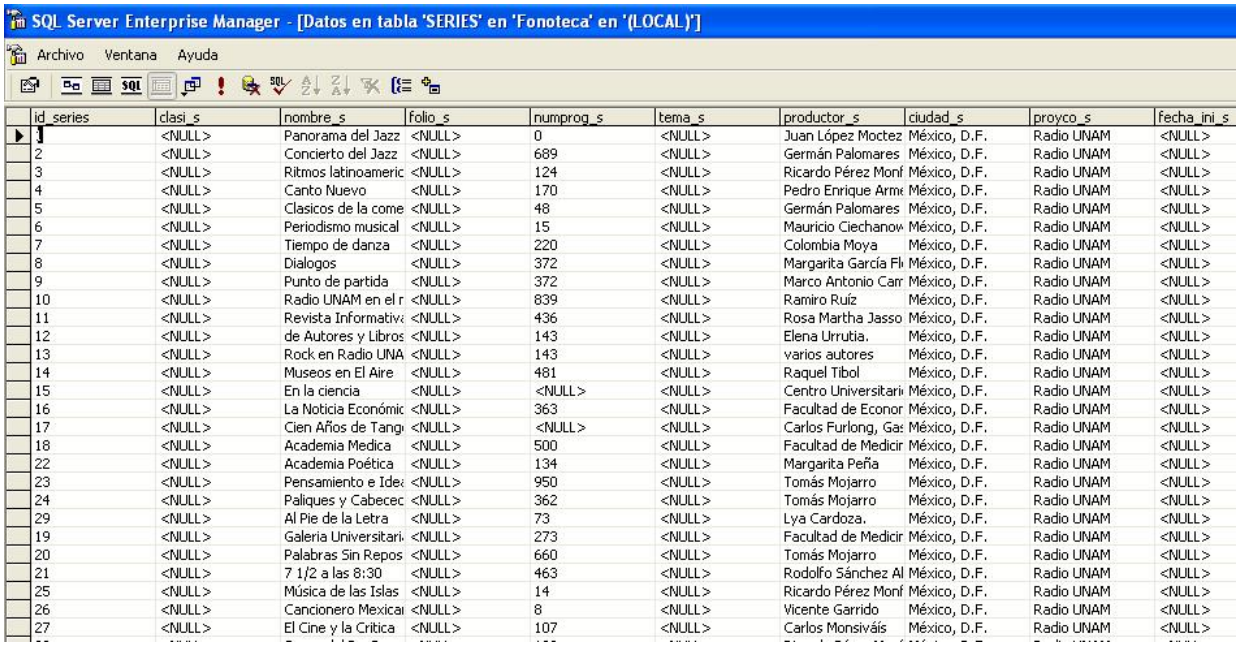


Figura 4.2.4 Creación del código SQL

Una vez generado el código se inserta en el analizador de consultas del servidor SQL Server y se genera la base de datos, con lo que se puede ahora insertar los datos correspondientes a cada una de las tablas. Algunos datos de series y programas se encuentran almacenados en hojas de cálculo (Excell, estos datos han sido recopilados por personas de servicio social que han laborado para Fonoteca), los cuales serán importados a la base de datos con la herramienta IMPORTAR datos del servidor SQL, lo que disminuirá el tiempo de captura de datos para el personal de Fonoteca. Después de importar los datos, el contenido de las tablas se puede visualizar como se muestra en la Figura 4.2.5.



id_series	clasi_s	nombre_s	folio_s	numprog_s	tema_s	productor_s	ciudad_s	proyco_s	fecha_ini_s
1	<NULL>	Panorama del Jazz	<NULL>	0	<NULL>	Juan López Moctez	México, D.F.	Radio UNAM	<NULL>
2	<NULL>	Concierto del Jazz	<NULL>	689	<NULL>	Germán Palomares	México, D.F.	Radio UNAM	<NULL>
3	<NULL>	Ritmos latinoamerica	<NULL>	124	<NULL>	Ricardo Pérez Monf	México, D.F.	Radio UNAM	<NULL>
4	<NULL>	Canto Nuevo	<NULL>	170	<NULL>	Pedro Enrique Arme	México, D.F.	Radio UNAM	<NULL>
5	<NULL>	Clasicos de la come	<NULL>	48	<NULL>	Germán Palomares	México, D.F.	Radio UNAM	<NULL>
6	<NULL>	Periodismo musical	<NULL>	15	<NULL>	Mauricio Ciechanow	México, D.F.	Radio UNAM	<NULL>
7	<NULL>	Tiempo de danza	<NULL>	220	<NULL>	Colombia Moya	México, D.F.	Radio UNAM	<NULL>
8	<NULL>	Dialogos	<NULL>	372	<NULL>	Margarita García Fl	México, D.F.	Radio UNAM	<NULL>
9	<NULL>	Punto de partida	<NULL>	372	<NULL>	Marco Antonio Cam	México, D.F.	Radio UNAM	<NULL>
10	<NULL>	Radio UNAM en el r	<NULL>	839	<NULL>	Ramiro Ruíz	México, D.F.	Radio UNAM	<NULL>
11	<NULL>	Revista Informativa	<NULL>	436	<NULL>	Rosa Martha Jasso	México, D.F.	Radio UNAM	<NULL>
12	<NULL>	de Autores y Libros	<NULL>	143	<NULL>	Elena Urrutia.	México, D.F.	Radio UNAM	<NULL>
13	<NULL>	Rock en Radio UNA	<NULL>	143	<NULL>	varios autores	México, D.F.	Radio UNAM	<NULL>
14	<NULL>	Museos en El Aire	<NULL>	481	<NULL>	Raquel Tibol	México, D.F.	Radio UNAM	<NULL>
15	<NULL>	En la ciencia	<NULL>	<NULL>	<NULL>	Centro Universitari	México, D.F.	Radio UNAM	<NULL>
16	<NULL>	La Noticia Económic	<NULL>	363	<NULL>	Facultad de Econor	México, D.F.	Radio UNAM	<NULL>
17	<NULL>	Cien Años de Tang	<NULL>	<NULL>	<NULL>	Carlos Furlong, Gas	México, D.F.	Radio UNAM	<NULL>
18	<NULL>	Academia Medica	<NULL>	500	<NULL>	Facultad de Medicir	México, D.F.	Radio UNAM	<NULL>
22	<NULL>	Academia Poética	<NULL>	134	<NULL>	Margarita Peña	México, D.F.	Radio UNAM	<NULL>
23	<NULL>	Pensamiento e Idea	<NULL>	950	<NULL>	Tomás Mojarro	México, D.F.	Radio UNAM	<NULL>
24	<NULL>	Paliques y Cabecec	<NULL>	362	<NULL>	Tomás Mojarro	México, D.F.	Radio UNAM	<NULL>
29	<NULL>	Al Pie de la Letra	<NULL>	73	<NULL>	Lya Cardoza.	México, D.F.	Radio UNAM	<NULL>
19	<NULL>	Galeria Universitari	<NULL>	273	<NULL>	Facultad de Medicir	México, D.F.	Radio UNAM	<NULL>
20	<NULL>	Palabras Sin Repos	<NULL>	660	<NULL>	Tomás Mojarro	México, D.F.	Radio UNAM	<NULL>
21	<NULL>	7 1/2 a las 8:30	<NULL>	463	<NULL>	Rodolfo Sánchez Al	México, D.F.	Radio UNAM	<NULL>
25	<NULL>	Música de las Islas	<NULL>	14	<NULL>	Ricardo Pérez Monf	México, D.F.	Radio UNAM	<NULL>
26	<NULL>	Cancionero Mexicar	<NULL>	8	<NULL>	Vicente Garrido	México, D.F.	Radio UNAM	<NULL>
27	<NULL>	El Cine y la Critica	<NULL>	107	<NULL>	Carlos Monsiváis	México, D.F.	Radio UNAM	<NULL>

Figura 4.2.5 Inserción de datos

Existe un método de creación en el que se establece cada uno de los campos de la tabla, asignándole el nombre y sus características, datos que contendrá, tamaño, descripción, formato, etc. La pantalla que presenta el programa para tal efecto es la que se ve en la figura 4.2.6.

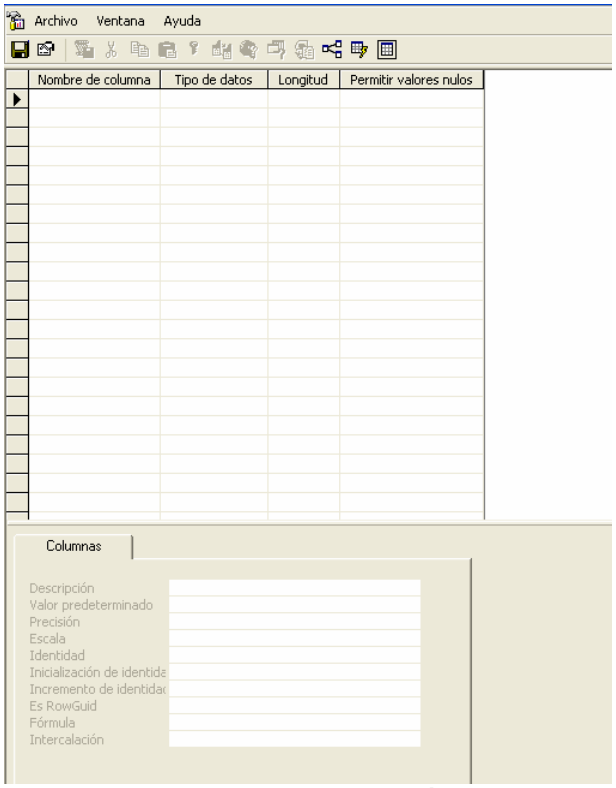


Figura 4.2.6 Pantalla para la creación de una tabla.

En la primera columna (nombre de columna) se escribe el nombre con el que se quiere identificar a dicho campo.

En tipo de datos**,** se selecciona de una lista disponible los datos que contendrá entre las siguientes opciones (figura 4.2.7):

Binary, char, datetime, decimal, float, int, money, real, text, varchar, por nombrar los más importantes.

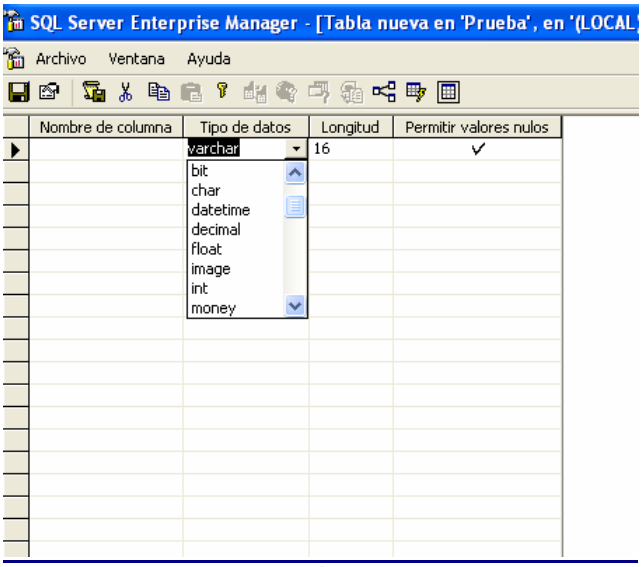


Figura 4.2.7 Elección del tipo de datos

Para ilustrar este proceso se muestra en la figura 4.2.8 la creación completa de la tabla señalando las relaciones con otras tablas y su llave primaria.

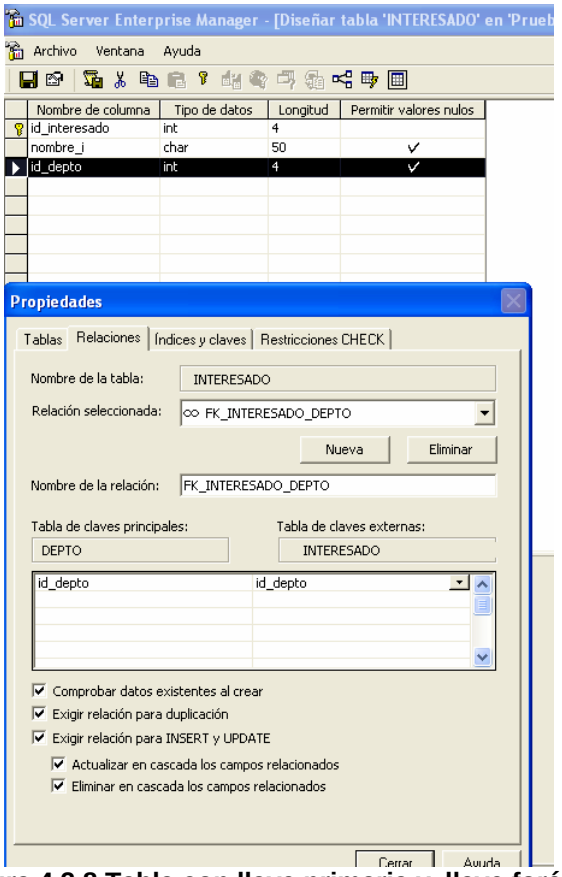


Figura 4.2.8 Tabla con llave primaria y llave foránea

Como el método para realizar las tablas fue por medio del código que se generó en Erwin no se incursionará más en esta forma de creación de las tablas.

4.3 Diseño y Construcción del Front End

Modos de Construcción de Pantallas.

En la herramienta elegida para el desarrollo del sistema, existen básicamente dos maneras de construir las pantallas que lo conforman:

- Form Wizar
- Manual

Form Wizar

Incluida dentro de Visual Basic 6.0 que permite generar formas mediante un asistente, que lleva paso a paso para la construcción de las formas. Esta herramienta tiene algunas ventajas y desventajas.

Como ventajas se puede mencionar que la generación de formas es más rápida y permite a desarrolladores con poca o nula experiencia hacer pantallas de sistemas que tengan interacción con bases de datos sin tener que introducir código.

Como desventajas se puede mencionar que la herramienta es poco flexible, es decir, no se les puede dar un aspecto personalizado o gráficamente atractivo a las formas que genera. Otra desventaja es la generación de mucho código que en ocasiones es necesario eliminar, dado que crea funcionalidad que pudiera no servir o no ser adecuada. Para los desarrolladores expertos, la utilización de esta herramienta se convierte más en un obstáculo que en una ayuda.

CAPÍTULO IV

Manual

La opción de crear las pantallas de forma manual es la más adecuada para el caso de este proyecto. El proceso para crear las pantallas de forma manual consiste de los siguientes pasos:

- Agregar los controles a la forma. Consiste en dibujar los controles necesarios a la forma que se va a realizar y que están disponibles en el cuadro de herramientas.
- Modificar las propiedades de los controles. Consiste en revisar las propiedades que están disponibles en la ventana de propiedades para un control específico y definir sus valores como son nombre, ancho, alto, color, estilo, etc.
- Escribir el código de los eventos adecuados. Consiste en programar las rutinas que se tienen que ejecutar y en los eventos de los controles que sean los más adecuados.
- Salvar la aplicación.

Menús y Pantallas del Sistema

El sistema que se desarrollará consta de diversas pantallas y menús. La forma de navegar entre estas pantallas y menús se explica en el grafico 4.3.1

A continuación se hará una breve descripción del objetivo de cada una de las pantallas.

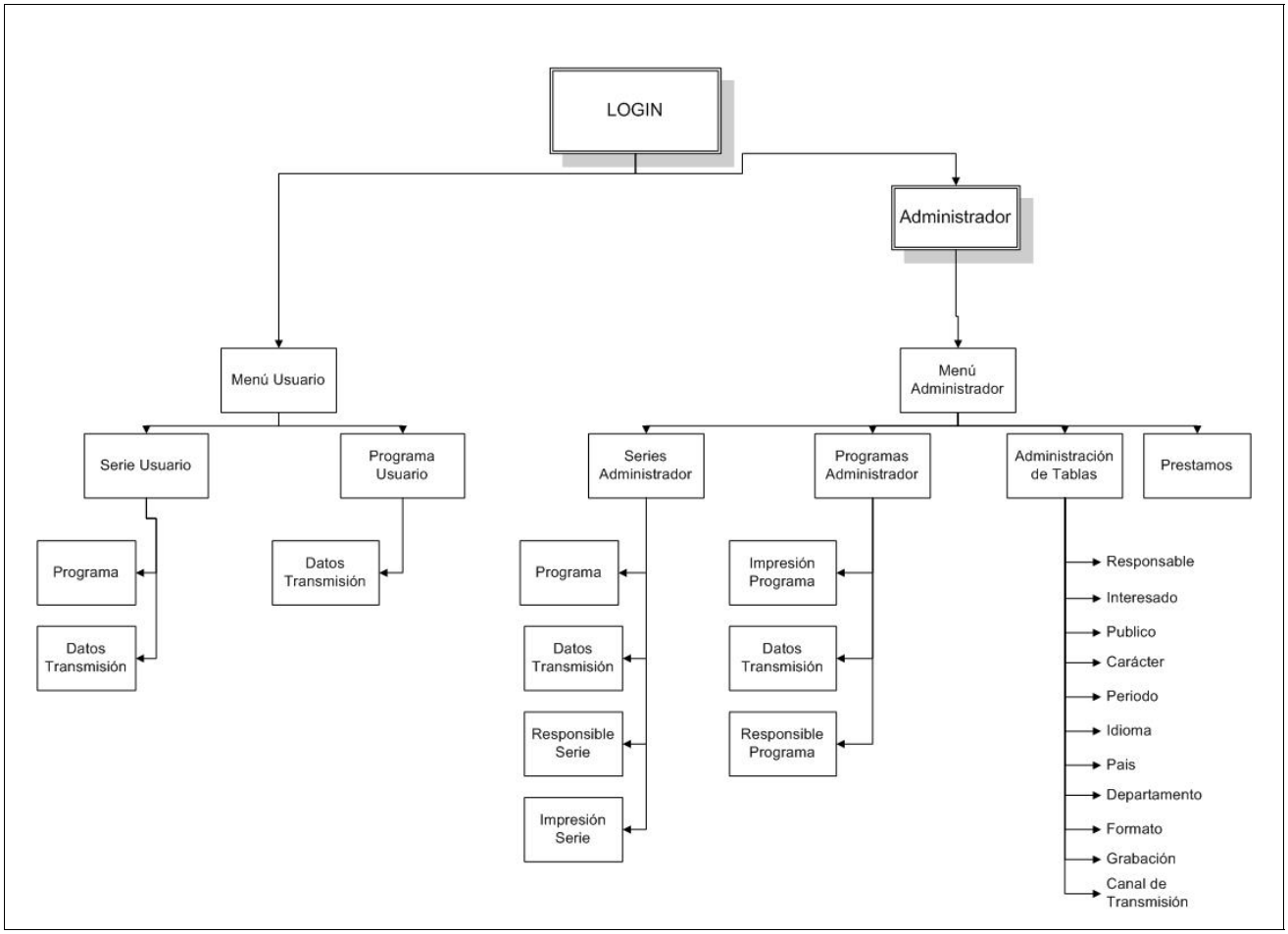


Figura 4.3.1 Diagrama de Menús y Pantallas

LOGIN. En esta pantalla, si es un administrador del sistema, inserta su login y su password para tener acceso a todas las utilidades, si no es administrador solo presiona el botón de “ENTRAR”.

Menú Usuario. Esta pantalla muestra los dos tipos de búsqueda que se pueden hacer, así como algunas opciones de éstas, como son las búsquedas por tema, invitado, conductor, etc.

Serie Usuario. El objetivo de esta pantalla es desplegar la información completa de la serie seleccionada.

- Programa. Desde la pantalla de Serie se puede acceder a la pantalla de Programa para ver los datos del programa seleccionado en la pantalla de Serie.

- Datos Transmisión. Esta pantalla muestra la fecha de la primera transmisión de la serie y sus retransmisiones, si es que tiene.

Programa Usuario. Se desplegará en esta pantalla los datos completos del programa seleccionado en la pantalla Menú Usuarios.

- Datos Transmisión. Despliega la información de la fecha de la primera vez que se transmitió el programa, y si fue retransmitido alguna vez.

Administrador. Esta pantalla solo despliega un mensaje de bienvenida con el nombre del administrador del programa.

Menú Administrativo. En esta pantalla se realizan las búsquedas de series y programas, además de la administración de otras tablas de la base de datos, y el préstamo del material de la fonoteca.

- La administración de las tablas de la base de datos consiste en la creación de nuevos campos y la actualización de campos ya existentes

Series Administrador. Se despliega la información en pantalla referente a la serie seleccionada, permitiendo la actualización de los datos.

- Programa Administrador. Se puede seleccionar un programa perteneciente a la serie para ver la información completa de éste.

- Canal Transmisión. Se ve el canal y la fecha de la primera transmisión, así como sus posibles retransmisiones, con la posibilidad de actualizar los datos.

- Responsable Serie. Se despliega la información del responsable de insertar los datos de la serie, y la persona responsable de la última modificación de los datos.

- Impresión Serie. Se muestra la vista previa de la impresión y la configuración de la impresora.

Programas Administrador. Se muestra en pantalla la información del programa seleccionado, permitiendo la actualización de los datos.

- Canal transmisión. La pantalla despliega la información de la fecha y canal de transmisión del programa, y si tiene retransmisiones, permitiendo la actualización de los datos.

- Responsable programa. En esta pantalla se muestran los datos de la persona encargada de insertar el programa y de la última persona que actualizó los datos.

- Impresión programa. Se despliega la vista previa del programa a imprimir, permitiendo la configuración de la impresora.

Controles

Para el desarrollo del sistema se emplearon los controles estándares de Visual Basic como cajas de texto, botones de comando, botones de opciones, etiquetas, etc. Y adicionalmente un control Active X desarrollado por terceros, que es DBRoports, y se utiliza para la construcción de los reportes de la aplicación.

Construcción del Front End

Se utilizará la pantalla de búsqueda administrativa para mostrar como se construyó la aplicación, tomando esta pantalla se demuestra como se construyeron las demás que forman el sistema.

El primer paso será crear el proyecto, el cuál será nombre “SFAGA” (Sistema Fonoteca Alejandro Goméz Arias) y será un proyecto estándar de Visual Basic (Figura 4.3.2)

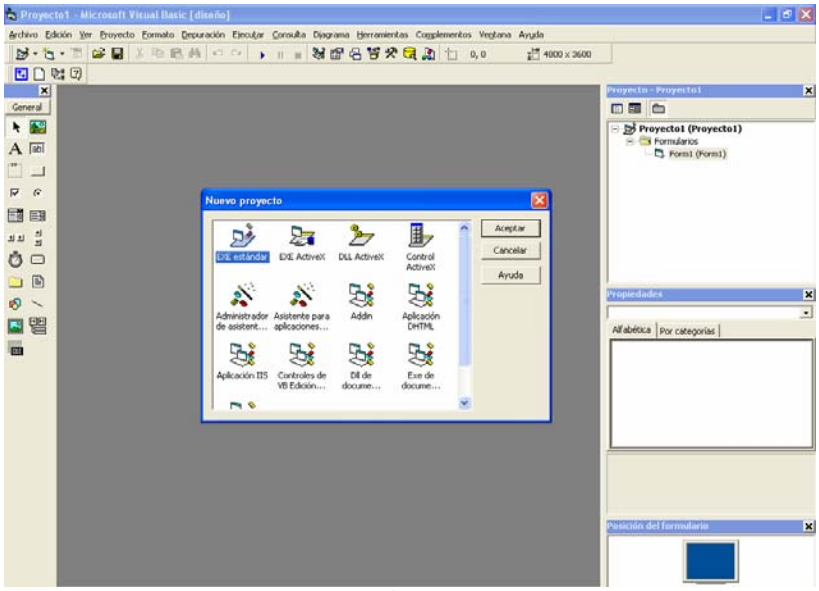


Figura 4.3.2 Selección del Tipo de Proyecto

Para crear la pantalla de búsqueda administrativa se colocará en una forma estándar un control Sstab con cuatro pestañas, donde se especificará que se puede realizar en cada una de ellas, en la primera pestaña se colocarán tres frames, los cuales servirán para separar las partes de la búsqueda, el primer frame contendrá tres botones de comando, tres etiquetas, una caja de texto y una caja combo, en el segundo frame se colocarán cinco etiquetas y una caja lista y en el último frame se pondrán dos botones de comando y una imagen. La forma tendrá las siguientes propiedades:

Nombre: fmrBusqueda_adm_s
Caption ”Título”: Sistema Fonoteca Alejandro Gómez Arias SFAGA RUNAM
Width “Ancho”: 12270
Heigth “Alto”: 9030
ControlBox “Cuadro de Controles”: False

Inicialmente, la forma se verá como se muestra en la Figura 4.3.3

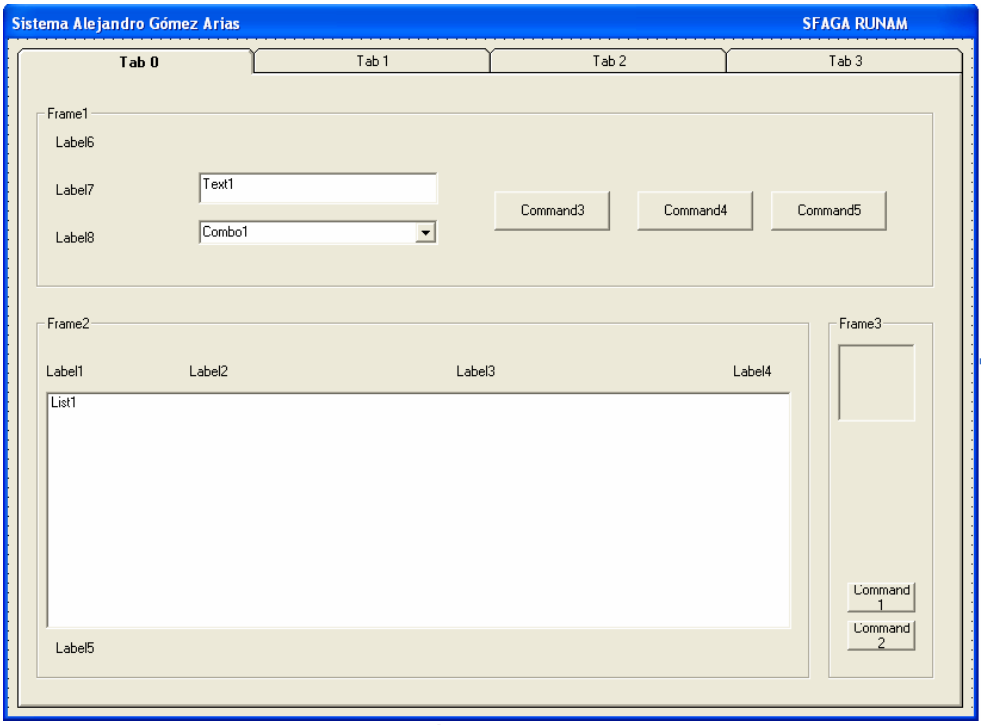


Figura 4.3.3 Búsqueda Administrativa

Se modificarán las propiedades de los botones de comando y tomarán los valores que se indican en la tabla a continuación:

Botón	Nombre	Titulo	Ancho	Alto	Estilo
Command 1	cmd_salir	SALIR	975	375	Estándar
Command 2	cmd_acerca	ACERCA	975	375	Estándar
Command 3	cmdBusqueda_s	CONSULTA	1455	495	Estándar
Command 4	cmd_verser	SERIES	1455	495	Estándar
Command 5	cmdSeries	LIMPIAR LISTA	1455	495	Estándar

Las propiedades de la caja combo son:

Nombre: cboBusqueda_s Ancho: 3015 Alto: 360

En las propiedades del Sstab se asignan los nombres de cada pestaña: BÚSQUEDA DE SERIES, BÚSQUEDA DE PROGRAMAS, ADMINISTRACIÓN DE TABLAS y PRÉSTAMOS.

Las etiquetas se muestran las características en la tabla:

Etiqueta	Título
Label 1	No. Serie
Label 2	Nombre de serie
Label 3	TOTAL DE REGISTROS ENCONTRADOS:
Label 4	No. Registros encontrados.
Label 5	Para ver la información correspondiente a cada serie, de DOBLE CLICK sobre el nombre de la SERIE
Label 6	Para realizar una consulta, proporcione la cadena a buscar, y en donde desea buscar, después haga click en CONSULTA. Para ver todas las SERIES registradas haga click en SERIES
Label 7	PALABRA A CONSULTAR:
Label 8	REALIZAR CONSULTA POR:

Los frames tomaron en su propiedad titulo CONSULTA POR SERIE, RESULTADO DE CONSULTA y el último frame tendrá vacía esta propiedad.

Una vez que se han realizado las modificaciones a los valores de las propiedades de la forma y controles, la pantalla se verá como se muestra en la figura 4.3.4

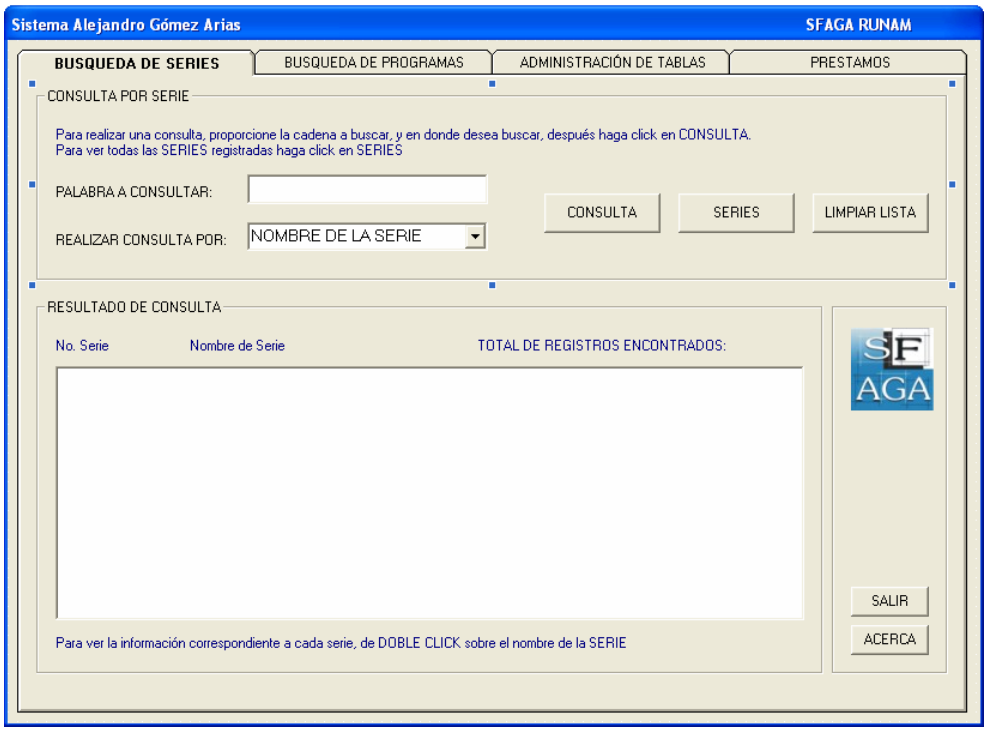


Figura 4.3.4 Búsqueda Administrativa (final)

Una vez terminado el diseño visual de la forma, se escribirá el código necesario para que al oprimir los botones, la caja combo y la lista se abran las pantallas correspondientes.

El primer procedimiento que se codificará es el evento Form_Load, el cual toma el nick del usuario ya que posteriormente se utiliza para la actualización de algunos campos.

```
Public Sub Form_Load()
 responsable = frmBienvenida.lblbenido 'se toma el nick del usuario válido
 optTabala(0).Value = True
 'se ocultan controles
 cmdAct_Ta.Visible = False
 cmbDep.Visible = False
 cmdCanNT.Visible = False
 refres_t.Visible = False
End Sub
```

Posteriormente se programarán los eventos click de los botones de comando para tener acceso a todas las funcionalidades del sistema.

```
Private Sub cmd_salir_Click(Index As Integer)
Me.Show
ojt = MsgBox("QUIERE SALIR DE SFAGA", 1 + 256 + 48, "SFAGA CERRAR ")
If ojt = 1 Then
 Unload Me
Else
 Me.Show
 Me.Enabled = True
End If
End Sub
```

El botón de comando cmd_acerca realiza las búsquedas que le especifica el combo desplegando los resultados en la lista.

```
Private Sub cmd_acerca_Click(Index As Integer)
fmrBusqueda_adm_s.Enabled = False
frmAcerca.Show
frmAcerca.cmd_Principal.Visible = True
End Sub
Private Sub cmdBusqueda_s_Click()
cvb = val_vacio(Trim(txtBusqueda_s.Text), "orv04")
If cvb = 0 Then 'primer if
cbv = MsgBox(" NO SE PROPORCIONADO UNA PALABRA A BUSCAR.", 48, "SFAGA BUSQUEDA
SERIES")
Else
lstBusqueda_s.Clear
lstBusqueda_s.Refresh
```

```
i = 0

Set rSERIES = New Recordset

'se captura campo de bsqueda

buspor = cboBusqueda_s.ListIndex

'se selecciona campo de busqueda

Screen.MousePointer = vbHourglass 'cursor de reloj de arena

Select Case buspor

 Case -1

  rSERIES.Open "select * from SERIE_BUSQUEDA WHERE (nombre_s like '%" &
Trim(fmrBusqueda_adm_s.txtBusqueda_s.Text) & "%') ORDER BY id_series", fCONEXION, adOpenKeyset,
adLockOptimistic, adCmdText

   Case 0

  rSERIES.Open "select * from SERIE_BUSQUEDA WHERE (nombre_s like '%" &
Trim(fmrBusqueda_adm_s.txtBusqueda_s.Text) & "%') ORDER BY id_series", fCONEXION, adOpenKeyset,
adLockOptimistic, adCmdText

   Case 1

  rSERIES.Open "select * from SERIE_BUSQUEDA WHERE (productor_s like '%" &
Trim(fmrBusqueda_adm_s.txtBusqueda_s.Text) & "%') ORDER BY id_series", fCONEXION, adOpenKeyset,
adLockOptimistic, adCmdText

   Case 2

  rSERIES.Open "select * from SERIE_BUSQUEDA WHERE (tema_s like '%" &
Trim(fmrBusqueda_adm_s.txtBusqueda_s.Text) & "%') ORDER BY id_series", fCONEXION, adOpenKeyset,
adLockOptimistic, adCmdText

   Case 3

  rSERIES.Open "select * from SERIE_BUSQUEDA WHERE (proyco_s like '%" &
Trim(fmrBusqueda_adm_s.txtBusqueda_s.Text) & "%') ORDER BY id_series", fCONEXION, adOpenKeyset,
adLockOptimistic, adCmdText

 Case 4

  rSERIES.Open "select * from SERIE_BUSQUEDA WHERE (clasi_s like '%" &
Trim(fmrBusqueda_adm_s.txtBusqueda_s.Text) & "%') ORDER BY id_series", fCONEXION, adOpenKeyset,
adLockOptimistic, adCmdText
```

```
Case 5
  rSERIES.Open "select * from SERIE_BUSQUEDA WHERE (tipo_pub_s like '%" &
Trim(fmrBusqueda_adm_s.txtBusqueda_s.Text) & "%') ORDER BY id_series", fCONEXION, adOpenKeyset,
adLockOptimistic, adCmdText
 Case 6
  rSERIES.Open "select * from SERIE_BUSQUEDA WHERE (vigencia_s like '%" &
Trim(fmrBusqueda_adm_s.txtBusqueda_s.Text) & "%') ORDER BY id_series", fCONEXION, adOpenKeyset,
adLockOptimistic, adCmdText
Case 7
  rSERIES.Open "select * from SERIE_BUSQUEDA WHERE (nombre_pais like '%" &
Trim(fmrBusqueda_adm_s.txtBusqueda_s.Text) & "%') ORDER BY id_series", fCONEXION, adOpenKeyset,
adLockOptimistic, adCmdText
End Select
If rSERIES.BOF = True Then 'no hay registros
lblResultado2_s.Caption = 0
Else
Do Until rSERIES.EOF
id = rSERIES("id_series")
If (id >= 0) And (id <= 9) Then
lstBusqueda_s.List(i) = rSERIES("id_series") & "                    " & rSERIES("nombre_s")
End If
If (id >= 10) And (id <= 99) Then
lstBusqueda_s.List(i) = rSERIES("id_series") & "                    " & rSERIES("nombre_s")
End If
If (id >= 100) And (id <= 999) Then
lstBusqueda_s.List(i) = rSERIES("id_series") & "                    " & rSERIES("nombre_s")
End If
If (id >= 1000) And (id <= 9999) Then
lstBusqueda_s.List(i) = rSERIES("id_series") & "                    " & rSERIES("nombre_s")
```

```
End If
 i = i + 1
 rSERIES.MoveNext
Loop
'se muestran el total de resultados
lblResultado2_s.Caption = lstBusqueda_s.ListCount
End If 'no hay registro
End If 'primer if
fraResultado_s.Enabled = True
lstBusqueda_s.Enabled = True
lblResultado1_s.Enabled = True
Screen.MousePointer = vbDefault 'restaura cursor
 End Sub
```

El botón de comando cmd_verser despliega en la lista todas las series existentes en la base de datos.

```
Private Sub cmd_verser_Click()
lstBusqueda_s.Clear
lstBusqueda_s.Refresh
i = 0
Screen.MousePointer = vbHourglass 'cursor de reloj de arena
Set rSERIES = New Recordset
'Call conexion
rSERIES.Open "select * from SERIE_BUSQUEDA ORDER BY id_series", fCONEXION, adOpenKeyset,
adLockOptimistic, adCmdText
Do Until rSERIES.EOF
id = rSERIES("id_series")
If (id >= 0) And (id <= 9) Then
```

```
lstBusqueda_s.List(i) = rSERIES("id_series") & "                    " & rSERIES("nombre_s")
End If
If (id >= 10) And (id <= 99) Then
lstBusqueda_s.List(i) = rSERIES("id_series") & "                   " & rSERIES("nombre_s")
End If
If (id >= 100) And (id <= 999) Then
lstBusqueda_s.List(i) = rSERIES("id_series") & "                  " & rSERIES("nombre_s")
End If
If (id >= 1000) And (id <= 9999) Then
lstBusqueda_s.List(i) = rSERIES("id_series") & "                 " & rSERIES("nombre_s")
End If
 i = i + 1
 rSERIES.MoveNext
Loop
'se muestran el total de resultados
lblResultado2_s.Caption = lstBusqueda_s.ListCount
fraResultado_s.Enabled = True
lstBusqueda_s.Enabled = True
lblResultado1_s.Enabled = True
Screen.MousePointer = vbDefault 'restaura cursor
End Sub
El botón de comando cmdSeries limpia la lista.
Private Sub cmdSeries_Click()
lstBusqueda_s.Clear
lblResultado2_s.Caption = lstBusqueda_s.ListCount
End Sub
```

El evento doble click de la lista permite ir a la forma en donde se desplegarán los datos completos de la serie seleccionada.

```
Private Sub lstBusqueda_s_DblClick()
 'bandera para cargar datos de db en la fmrPresentar_adm_s
 ban1 = 1
 frmPresentar_adm_s.Show
 frmPresentar_adm_s.Refresh 'refresca para que se redibuje el formulario
 frmPresentar_adm_s.fra_ins_s.Visible = False
End Sub
```

4.4 Pruebas e Integración del Sistema

Objetivos de la pruebas

El objetivo de la prueba es descubrir algún error, y siendo buena cuando su ejecución conlleva una probabilidad elevada de encontrar un error que hasta el momento no ha sido detectado. El éxito de la prueba se mide en función de la capacidad de detectar un error que estaba oculto.

Existen fundamentalmente dos enfoques: prueba de la caja blanca y prueba de la caja negra.

Pruebas de la caja blanca

La prueba de la caja blanca usa la estructura de control del diseño procedural para derivar los casos de prueba. La idea es confeccionar casos de pruebas que garanticen la verificación de todos los caminos llamados independientes; que estos caminos independientes se prueben en sus dos facetas desde el punto de vista lógico, es decir, verdadero y falsa, se ejecuten todos los bucles en sus límites operacionales y ejerciten las estructuras internas de datos para asegurar su validez.

Este tipo de pruebas no son muy útiles para probar aplicaciones desarrolladas con Visual Basic, ya que Visual Basic más que crear líneas de código lo que hace es crear formularios para introducir datos. Por lo tanto, se las pruebas utilizadas para esta aplicación son las caja negra, que se describen a continuación.

Pruebas de la caja negra

Los métodos de la caja negra enfocan los requisitos funcionales del software permitiendo disponer de conjuntos de valores de entrada que ejerciten de forma completa todos los requisitos del programa.

No es una alternativa a la prueba de caja blanca, sino que se complementan, ayudando a encontrar diferentes tipos de errores. La prueba de la caja negra intenta encontrar errores de los siguientes tipos fundamentales:

- Funciones incorrectas o inexistentes.
- Errores relativos a las interfaces.
- Errores en estructuras de datos o en acceso a las bases de datos externas.
- Errores debidos al rendimiento.
- Errores de inicialización o terminación.

La técnica de prueba de la caja negra forma parte en gran medida de las pruebas de integración del software pues se ignora la estructura de control.

Pruebas en Visual Basic y SQL Server

Este proyecto desarrollado fue realizado con Visual Basic, el cual es un programa para crear aplicaciones en bases de datos, por lo que es de gran importancia comprobar que las estructuras de los datos (tablas de SQL) creadas para mantener la base de datos son las adecuadas.

Básicamente se centro en probar los formularios y los eventos que poseen los componentes de dichos formularios.

Principalmente, Visual Basic crea aplicaciones que no son puro código, sino que crea formularios que son una especie de interfaz con el usuario. Por tanto, las pruebas de caja blanca, que examinan los posibles caminos en la ejecución del código, no probaron completamente la aplicación realizada, se utilizó la prueba de la caja negra, para probar los formularios mediante diversos datos de entrada que comprobaron si el programa funcionaba como se deseaba. Es decir, aceptando los datos correctos y rechazando los datos no válidos según la especificación del programa.

Otro aspecto que se probó son los eventos que poseen los componentes de cada formulario. Un evento es una acción que puede ocurrir sobre un componente, como al pulsar un botón surge un evento que ejecuta un código asociado a este. Por lo tanto, se comprobó para cada componente del formulario y el evento asociado a él, se ejecutará el código adecuado.

En el formulario de Serie Administrativa (Figura 4.4.1), se permite la actualización de la serie que se encuentra en ese momento en pantalla, así como la posibilidad de ver otras pantallas que despliegan información de otros datos, como lo son los programas, el responsable de la serie y las fechas en que se transmitió la misma, agregando los cambios realizados en la base de datos.

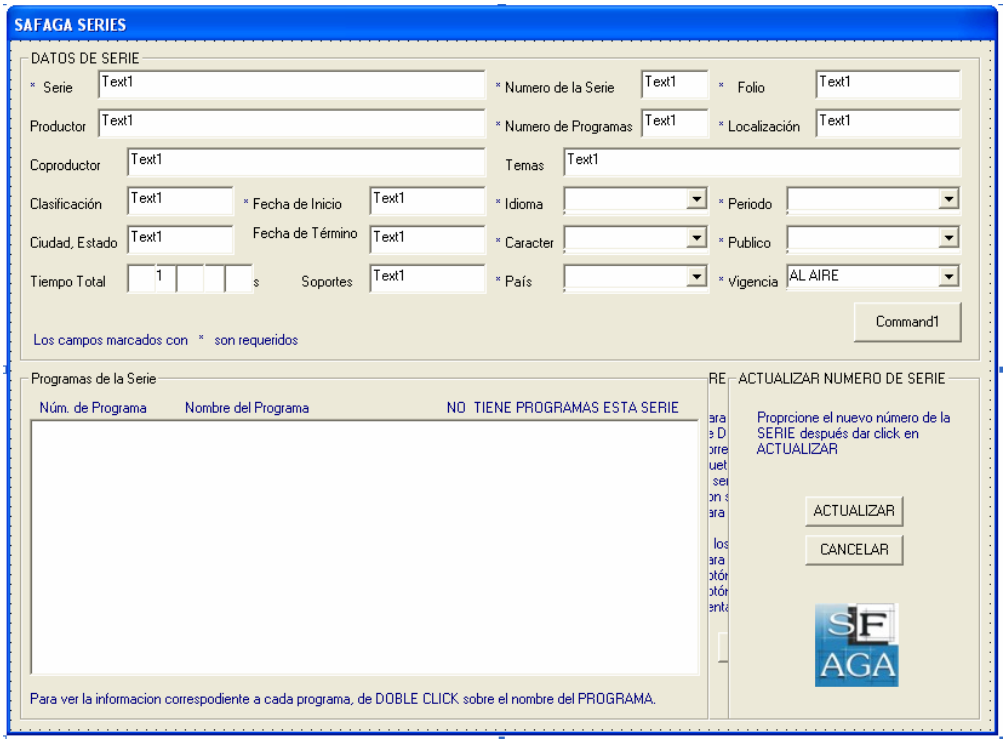


Figura 4.4.1 Formulario Serie Administrativa

La primera prueba que se realizó es cuando al insertar una nueva serie, los datos agregados deben ser válidos para cada caja de texto, es decir, que la aplicación no admita valores alfabéticos en donde corresponden números, como son las fechas o el tiempo de la serie.

También se probó que en las listas combo no se pudiera introducir información, para que solo se puedan seleccionar las opciones ya previstas.

Se probó el botón “Regresar” para ver que desplegara la pantalla adecuada, para probar el botón “Aceptar “, como esta asociado con el objeto opciones, se probó que ocultará y mostrará los marcos adecuados a cada opción.

Para la opción “Actualizar Datos” aparecen dos botones, “Cancelar” que oculta las listas combo y muestra las cajas de texto con la información desplegada anteriormente, ocultando los dos botones antes mostrados. El segundo botón, “Actualizar” valida que se hallan insertado mínimo los campos requeridos, como son el nombre y número de la serie, número de programas, localización, folio y fecha de inicio, así como que los datos insertados sean válidos y correctos entonces se actualiza la base de datos.

En la opción “Actualizar número de Serie” aparecen dos botones, Cancelar y Aceptar, el primero es general para todas los opciones, regresar los datos sin modificaciones y ocultar los botones. El botón Aceptar en este caso coloca el promt en el campo “Número de Serie” y cambia el color del fondo para indicar al usuario que dato es el que se debe modificar, sí el dato ya existe se manda un mensaje informando de su existencia, de lo contrario se actualiza la base de datos.

La opción “Agregar Nuevo Programa” oculta la forma actual y muestra en pantalla la forma de “Programa Administrativo” con los campos vacíos listos para insertar los datos de un nuevo programa perteneciente a la serie de la cual se procede. (Figura 4.4.2)

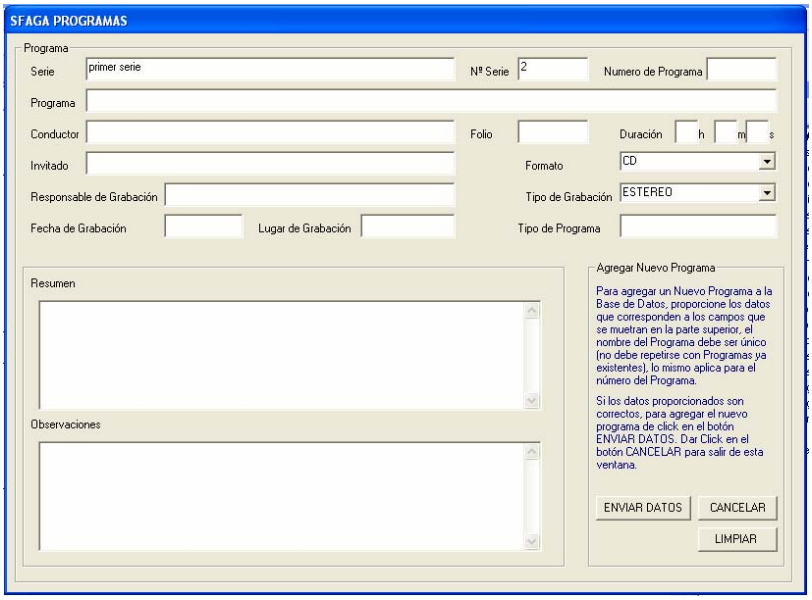


Figura 4.4.2 Nuevo Programa

Con la opción “Agregar Nueva Serie” todos los campos se ponen en blanco para insertar una nueva serie, mostrando tres botones, nuevamente el botón “Cancelar”, el botón “Limpiar”, que se encarga de limpiar todos los campos y el botón “Enviar Datos”, el cual inserta a la base de datos la nueva serie, verificando los datos requeridos y si son válidos. (Figura 4.4.3)



Figura 4.4.3 Nueva Serie

"Datos de Transmisión" muestra una nueva pantalla que contiene las fechas en que se ha transmitido la serie, así como el canal de transmisión y la hora, con la opción de modificar estos datos para las retransmisiones, no así en la primera vez que fue transmitida la serie. (Figura 4.4.4)

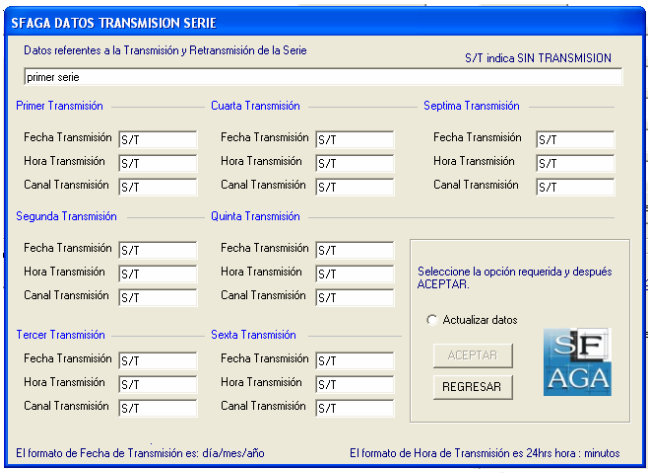


Figura 4.4.4 Datos Transmisión

La opción "Resumen de la Serie" muestra una pantalla con el resumen de la serie, permitiendo actualizarlo.

"Responsable de Serie" despliega una pantalla que muestra los datos de la persona encargada de insertar la serie en la base de datos, así como la última persona que realizó cambios en la

información de la serie. (Figura 4.4.5)



Figura 4.4.5 Responsable Serie

Por último se verificó que la acción de dar doble click sobre un programa perteneciente a la serie, mostrados en la lista, lleve a la pantalla de programas y contenga los datos adecuados de dicho programa.

Pruebas de Implantación del Sistema.

Se realizan las pruebas de implantación del sistema con el fin de comprobar el funcionamiento correcto del mismo en el entorno de operación y permitir al usuario, que determine la aceptación del sistema una vez instalado en su entorno real, según el cumplimiento de los requisitos no funcionales específicos.

Una vez ejecutadas estas pruebas, se informa de las incidencias detectadas al responsable de implementación, el cual analiza la información y toma las medidas correctivas que considere necesarias para que el sistema de respuesta a las especificaciones previstas, momento en el que se da por probado.

Desarrollo

El sistema se ve sometido al siguiente plan de actualización:

- Recuperación entre fallos del sistema. Se procede a insertar datos erróneos en cada uno

de los elementos lógicos del programa, en la parte de nueva serie, actualización de serie, nuevo programa y actualización del programa, así como en las tablas menores.

- Errores de tipo. Ejemplo: introducir cadenas de texto donde se pide un valor numérico.
- Duplicación de las llaves primarias. Introducir datos repetidos que sólo pueden aparecer de forma unívoca.
- Introducción de información inconsistente.
- Sustituciones inesperadas. Forzar actuaciones atípicas para tratar de ver fallos del sistema.
- Seguridad. Se comprueba que los métodos de seguridad funcionan frente a intentos de accesos no permitidos.
- Rendimiento. Se sobrecarga al sistema con exceso de información o tareas para comprobar que funciona de una forma estable.

La realización de las pruebas ha sido llevada con éxito, sin hallar fallo alguno del sistema, siendo todas las respuestas esperadas.

Pruebas de Aceptación del Sistema.

Las pruebas de aceptación se llevan a cabo con el fin de validar que el sistema cumple los requisitos básicos de funcionamiento esperado y permitir al usuario que determine la aceptación del sistema. Por este motivo, estas pruebas son realizadas por el usuario final del sistema y es durante este periodo de tiempo, cuando debe plantear todas las deficiencias o errores que encuentre antes de dar por aprobado el sistema definitivamente.

Desarrollo

Se elabora un test para que el usuario final realice una serie de comprobaciones a lo largo de un periodo de tres meses, tratando de hallar posibles anomalías en el funcionamiento de la aplicación.

El plan de pruebas consiste en pequeños pasos y fórmulas a fin de verificar que las respuestas esperadas se cumplen:

- Comprobar que el ingreso de nuevas series y nuevos programas es correcto.
- Comprobar que las actualizaciones de las series y los programas se realizan correctamente.
- Comprobar que las búsquedas se realizan correctamente.
- Comprobar que la inserción de nuevos datos en las tablas menores se realice correctamente.
- Generar los reportes y comprobar que salen todos los datos requeridos.

4.5 Factibilidad Operativa

La factibilidad de un sistema se refiere a que se investiguen diferentes opciones técnicas y sus consecuencias, se planifique un programa principal de plazos y recursos y se evalúen los potenciales riesgos en el proyecto.

Factibilidad técnica.

Es aquí donde se estudia si el trabajo para el proyecto, puede desarrollarse con el software, hardware y el personal existente, y en caso de necesitar nueva tecnología, cuáles son las posibilidades de desarrollarla.

Para la realización del proyecto se usó una herramienta de cuarta generación como Visual Basic 6.0 y se implementaron las tablas de la base de datos con SQL Server. Esto supone una factibilidad operativa ya que la aplicación tiene una Interfaz Gráfica de Usuario (GUI) fácil de utilizar y una presentación de Windows XP.

Esto conlleva un tiempo de aprendizaje muy pequeño por parte del usuario para que este pueda manejar y dominar completamente el sistema.

La aplicación se ha diseñado para que sea fácil de modificar en el caso de actualizaciones en el futuro.

El sistema realizado debe ser ejecutado en entorno Windows y funciona bajo Windows 98 o versiones superiores de este mismo sistema operativo.

Requisitos del software

- SQL Server en el servidor
- Windows 98 o superior

Requisitos de hardware

- Procesador Pentium o superior
- 32 MB de memoria RAM
- Monitor VGA

Instalación del Software

- Para instalar Visual Basic o SQL Server se siguen los siguientes pasos:
- Iniciar Windows
- Insertar el disco de Setup “Instalación” del software en la unidad de CD-ROM, regularmente D:.
- Desde el administrador de programas de Windows (Figura 4.5.1) seleccionar Run “Ejecutar” del menú File “Archivo”.

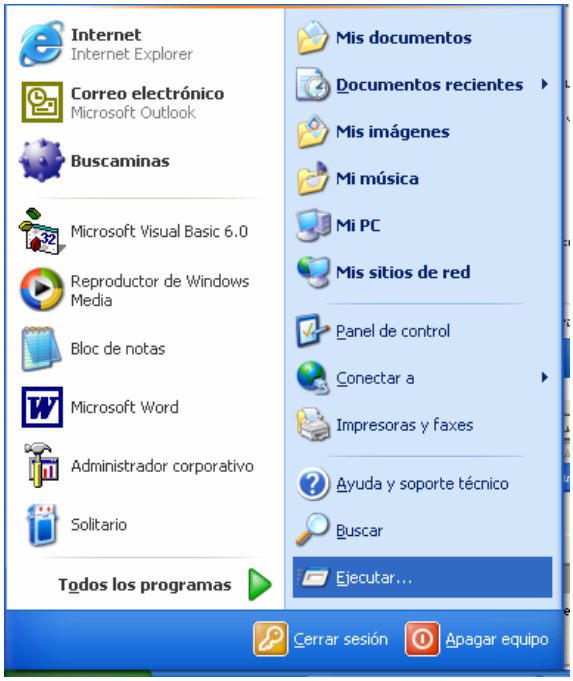


Figura 4.5.1 Administrador de programas Windows

- En la caja de línea de comandos (Figura 4.5.2), teclear D:\setup.

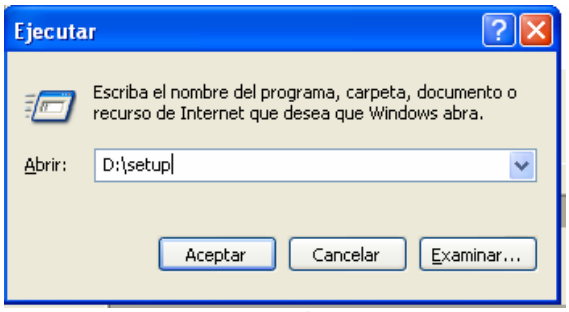


Figura 4.5.2 Caja de línea de comandos

- Escoger el botón OK “Aceptar”.
- Cuando se inicie Setup, seguir las instrucciones de las pantallas.

Factibilidad operativa

En esta fase se investiga si a los usuarios les será de utilidad el sistema, y si realmente se usará, si habrá algún tipo de capacitación y si se le tendrá que dar algún tipo de mantenimiento al sistema.

CAPÍTULO IV

Plan de Pruebas

La definición del plan de pruebas, el cual sirve como guía para la realización de las pruebas y permite verificar que el sistema de información cumpla las necesidades establecidas por el usuario. El plan se inicia en el proceso Análisis del Sistema, definiendo el marco general, y estableciendo los requisitos de prueba de aceptación, relacionados directamente con la especificación de requisitos.

Dicho plan se irá complementando y detallando a medida que se avanza en los restantes procesos del ciclo de vida del software, diseño del sistema, construcción del sistema e implementación y aceptación del sistema.

Se plantean los siguientes niveles de prueba:

- Pruebas unitarias
- Pruebas de integración
- Pruebas del sistema
- Pruebas de implantación
- Pruebas de aceptación

Ejecución de las pruebas unitarias

En esta actividad se realizan las pruebas unitarias de cada uno de los componentes del sistema, una vez codificados, con el objeto de comprobar que su estructura es correcta y que se ajusta a la funcionalidad establecida.

Desarrollo

Se dispone de una computadora donde se ejecutarán las pruebas sobre cada módulo de forma independiente. Como la fonoteca “Alejandro Gómez Arias” cuenta con una base de datos, se tomará una pequeña muestra para así probar la aplicación de manera más realista.

Se seguirá un plan de pruebas consistente en la ejecución de todos los pasos básicos de cada módulo, introduciendo información de prueba para ver si se reflejan de forma adecuada en la base de datos y las posteriores acciones son administradas según lo permitido en cada estado de funcionalidad, como por ejemplo, que la base de datos impida dar de alta a un usuario ya existente.

Realización de las pruebas unitarias

Los cambios en la base de datos han sido los esperados, es decir, todo ha funcionado correctamente. Tras realizar las pruebas se ha examinado cada una de las tablas de la base de datos para comprobar que todo es correcto.

En cuanto a la funcionalidad permitida en cada estado de ejecución, también ha tenido éxito, mostrándose en cada caso las ventanas adecuadas con la información correspondiente a cada situación.

Ejecución de las pruebas de integración

El objeto de las pruebas de integración es verificar si los componentes o subsistemas interactúan correctamente a través de sus interfaces, tanto internas como externas, cubren la funcionalidad establecida y se ajustan a los requisitos no funcionales especificados en las verificaciones correspondientes.

Esta actividad se realiza en paralelo a las actividades Generación del Código de los Componentes y Procedimientos y Ejecución de las Pruebas Unitarias. Sin embargo, es necesario que los componentes objeto de las pruebas de integración se hayan verificado de manera unitaria.

Desarrollo

Se dispone de una computadora donde se integrarán todas las ventanas de la aplicación. Cada una de ellas se enlazará a las opciones de la ventana principal, y a través de ello a otras ventanas, intercambiando información o tan solo invocando su apertura con una relación de tipo modal.

El plan de pruebas consistirá en comprobar:

- La correcta apertura de ventanas según las conexiones establecidas.
- El correcto paso de información entre ventanas.
- La consistencia de estados entre ventanas al ejecutar determinadas funciones.

Realización de la pruebas de integración

- Aceptación de ventanas asociadas a cada acción.
- Datos generados en la transición entre dos ventanas distintas.
- Estados de cada ventana al variar los estados de una ventana relacionada.

Prueba del sistema

Las pruebas a este tipo de programas que funcionan con distintas ventanas y se activan con botones (por eventos), representa una revisión final de las especificaciones, del diseño y de la implantación.

Mantenimiento

El mantenimiento está asociado a la corrección de errores, a nuevas adaptaciones requeridas por la evolución del entorno del software, así como los cambios y nuevas especificaciones propuestas por el cliente.

Mientras que el cambio tecnológico afecta indirectamente a los sistemas en el software, el entorno de trabajo y los usuarios lo hacen directamente, produciendo demandas de mantenimiento adaptativo y perfectivo respectivamente.

Mantenimiento correctivo

A pesar de las pruebas y verificaciones que aparecen en etapas anteriores del ciclo de vida del software, los programas pueden tener defectos. El mantenimiento correctivo tiene por objetivo

localizar y eliminar los posibles defectos de los programas.

Mantenimiento adaptativo

Este tipo de mantenimiento consiste en la modificación de un programa debido a cambios en el entorno (hardware o software) en el cual se ejecuta.

Mantenimiento perfectivo

Cambios en la especificación, normalmente debidos a cambios en los requerimientos de un producto de software, implican un nuevo tipo de mantenimiento llamado perfectivo. Se puede definir el mantenimiento perfectivo como el conjunto de actividades para mejorar o añadir nuevas funcionalidades requeridas por el usuario.

Mantenimiento preventivo

Este último tipo de mantenimiento consiste en la modificación del software, para mejorar las propiedades de dicho software sin alterar sus especificaciones funcionales.

4.6 SQL Server

Por cuestiones de seguridad se decidió que las tareas de eliminar y realizar respaldos, serán hechos únicamente por el administrador del Servidor de base de datos SQL Server (dba), por tal razón se mencionaran a continuación las pasos necesarios para realizar sendas tareas.

Eliminar de la base de datos.

Para eliminar se debe tener en cuentan que existen dependencias entre los datos, por ejemplo, no se puede eliminar una serie sin antes eliminar los programas que pertenecen a dicha serie.

El primer paso es abrir la tabla donde se encuentra el dato que se desea eliminar (Figura 4.6.1).

Esto se hace desde la consola de administración del Servidor SQL Server Enterprise Manager[3].

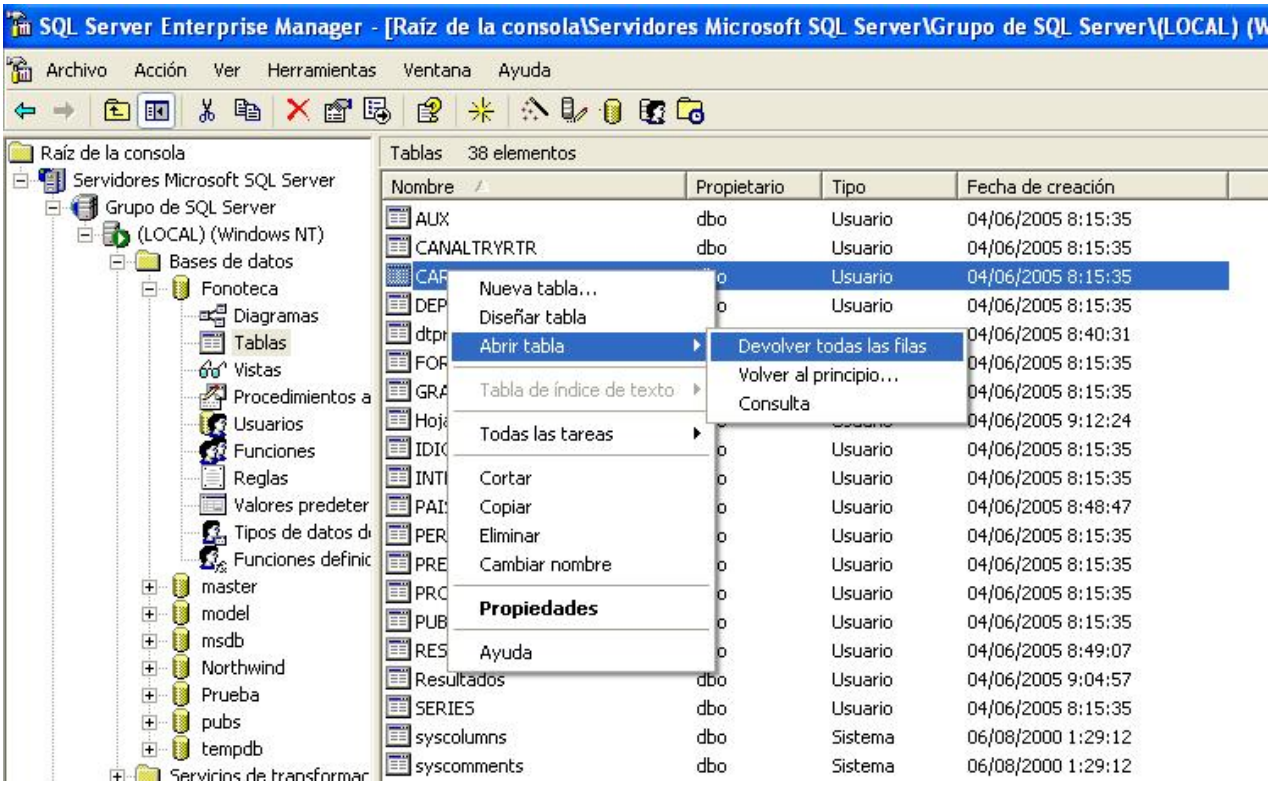


Figura 4.6.1 Selección de tabla de la BD

Una vez que se desplegaron todos los datos que contiene la tabla, se selecciona el dato que se quiere eliminar (Figura 4.6.2) y se oprime la tecla “Supr” o “Del”

[3] Al cual se accede desde: Inicio→Programas→SQL Server→Administrador Corporativo

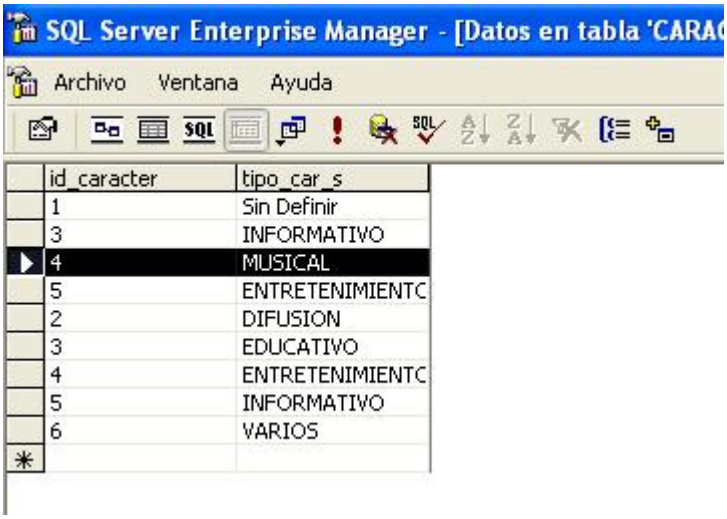


Figura 4.6.2 Eliminar el dato elegido

Si el dato que se eligió depende de otros valores no se permitirá la eliminación de este dato. EL procedimiento anterior funciona para cualquiera de las tablas de la base de datos, sin embargo, si requiere eliminar una serie con todos sus programas, todos los programas de una serie, o un rango de programas pertenecientes a una serie, estas tareas se realizan con instrucciones sql las cuales se ejecutan desde el analizador de consultas de SQL Server (Ver manual de usuario). Para que el administrador del servidor SQL Server pueda eliminar un registro, tiene que recibir la petición escrita y autorizada por los jefes de los departamentos de fonoteca y cómputo/sistemas, esta petición se almacena en ambos departamentos para aclaraciones.

Respaldo de la base de datos (backup).

Para realizar los respaldos de la base da datos se utiliza el plan de mantenimiento que tiene la aplicación (SQL Server) (Figura 4.6.3), este plan de mantenimiento se lanza con un click derecho sobre la base datos Fonoteca.

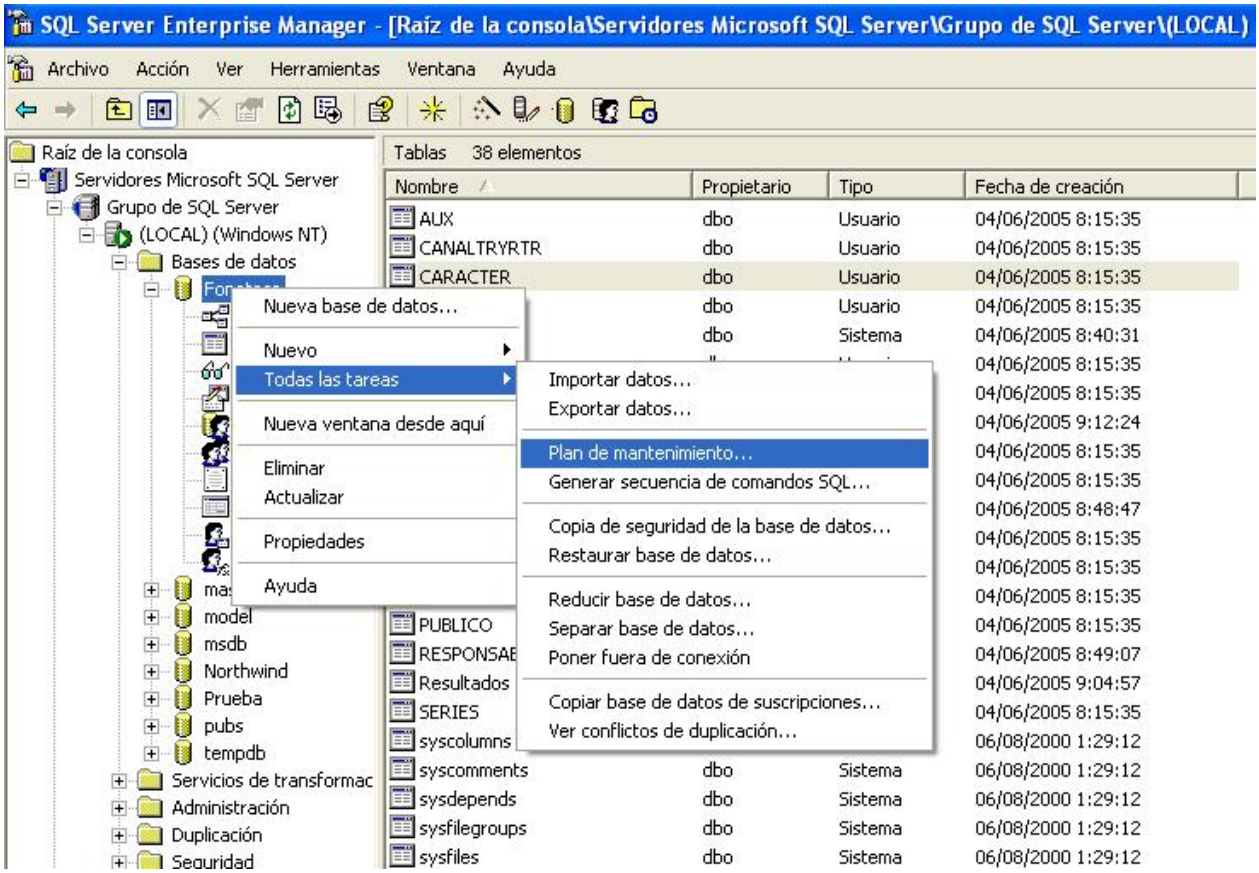


Figura 4.6.3 Plan de Mantenimiento

El cual despliega un Wizard muy sencillo que permite programar las fechas y los lugares de almacenamiento de los respaldos de la base de datos (Figura 4.6.4).

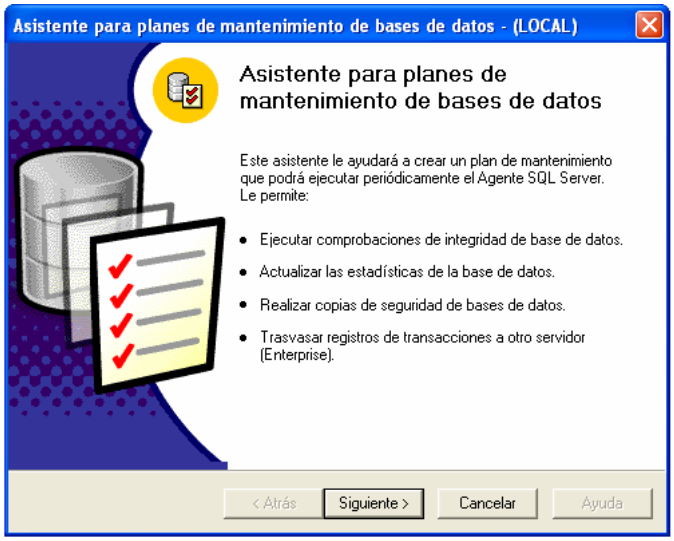


Figura 4.6.4 Wizard de respaldo de la BD

En la Figura 4.6.5 se muestra los parámetros que le permiten editar los respaldos, estos parámetros son el día de la semana, la hora, el día del mes en que se desea realizar el respaldo, por mencionar las más importantes.

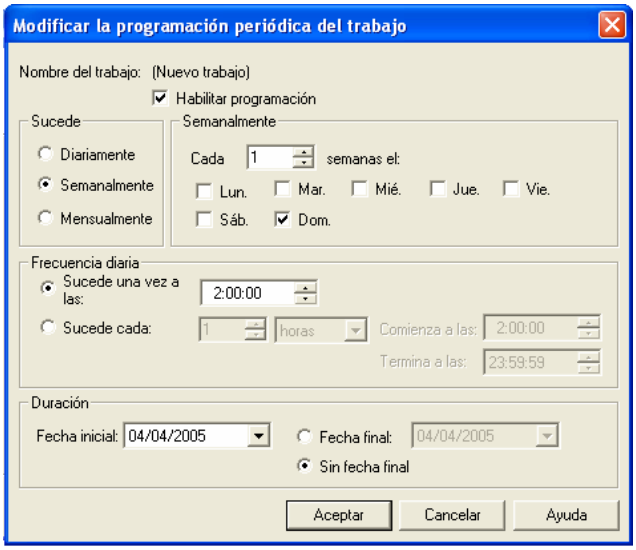


Figura 4.6.5 Configuración de parámetros

Una vez programado el respaldo (copia de seguridad de base de datos), este se realizará en fecha y hora programadas, este respaldo tendrá una extensión BAK. Para que se realice el respaldo se debe asegurar que el Agente SQL Server esta activado. Figura 4.6.6

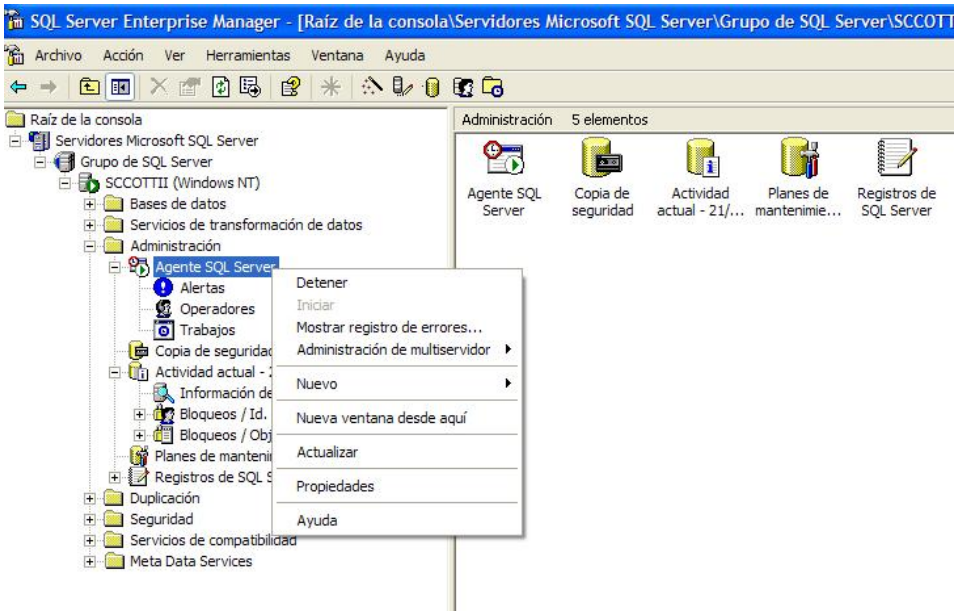


Figura 4.6.6 Agente SQL Server

Recuperar respaldo de la base de datos (restore).

Para recuperar la base de datos desde un previo respaldo, al igual que el respaldo de la base de datos se utiliza el plan de mantenimiento que tiene la aplicación (SQL Server) (Figura 4.6.3), eligiendo la opción “Restaurar base de datos”, con lo que se despliega un Wizard en el que se indica el respaldo del cual se hará la recuperación de la base de datos (Figura 4.6.7).

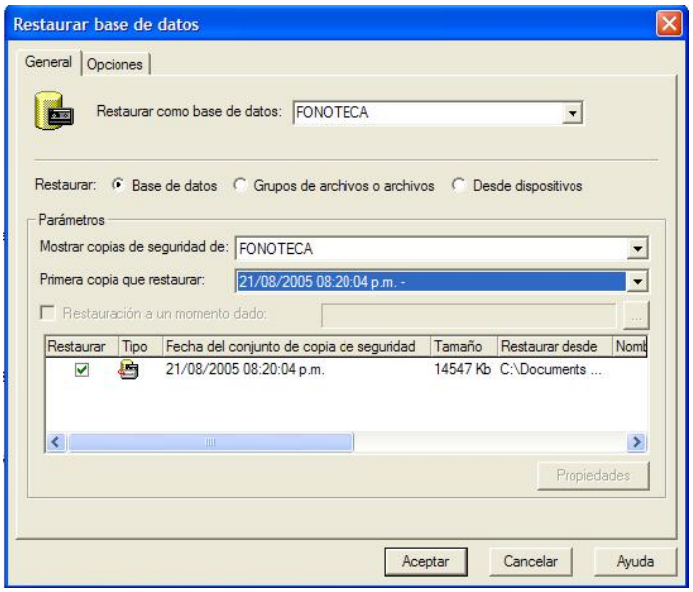


Figura 4.6.7 Agente SQL Server

Configuración de la Red

SFAGA es una aplicación que permite que varios usuarios estén trabajando al mismo tiempo, para configurar esta opción se sigue desde el del sistema operativo Panel de Control, Herramientas Administrativas, Orígenes de datos ODBC (Figura 4.6.8)

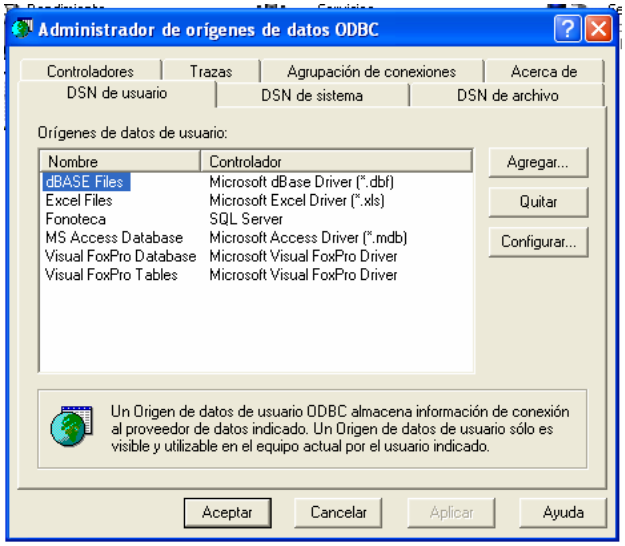


Figura 4.6.8 Administrador de Orígenes de Datos ODBC

En esta parte se agrega el DNS del usuario, especificando el origen de los datos, en este caso SQL Server, y se pedirá que se le de un nombre al origen de los datos con el cual se identificara, así como el servidor al que se desea conectar.

Para poder hacer uso de los datos se pide una forma de autentificación, puede ser por medio de la autentificación de Windows o con la de SQL, una vez que se especificó el modo de autentificación se pide especificar una base de datos como predeterminada, siendo en esta opción la que almacena los datos (Fonoteca), para finalizar se pide el idioma y si la conexión debe cifrar los datos al transmitir.

Para permitir comunicación con los clientes, el servidor debe aceptar las transacciones vía TCP-IP (Figura 4.6.9)

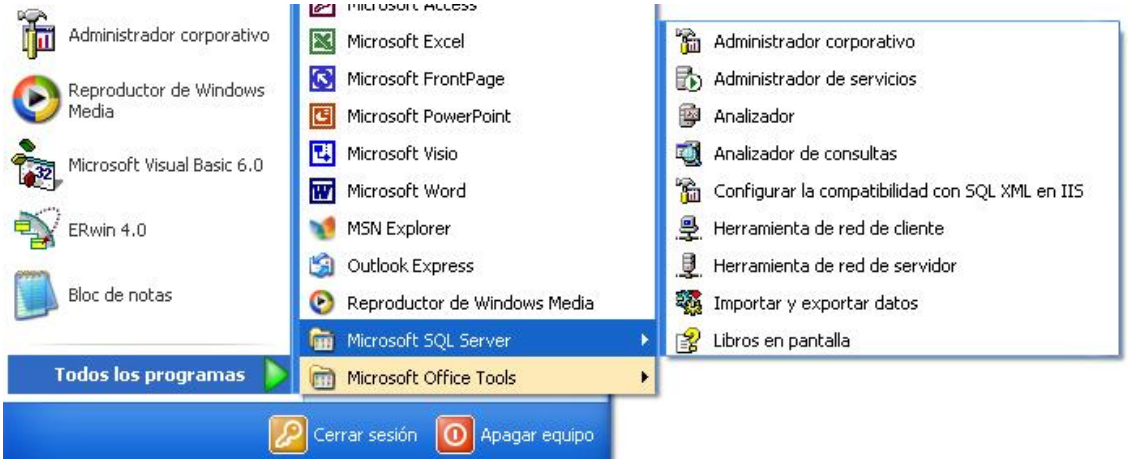


Figura 4.6.9 Server SQL TCP/IP 1

Dirigirse desde “Inicio” a la aplicación Microsoft SQL Server – Herramientas de red de servidor, lo cual desplegará la siguiente ventana. Figura 4.6.10



Figura 4.6.10 Herramientas de Red SQL Server

En esta ventana se configura el protocolo de comunicación, en la parte protocolos habilitados se habrán de localizar canalizaciones con nombre y TCP/IP. Una vez que se tengan estos protocolos habilitados, se acepta la configuración y el Server SQL esta habilitado para aceptar peticiones del cliente.

CONCLUSIONES

5.1 Conclusiones

Dado que una de las principales labores de la fonoteca “Alejandro Gómez Arias” es preservar el material grabado, catalogado, distribuido y en condiciones adecuadas para poder difundirlo, además de realizar el manejo rápido y adecuado de toda la información que se tiene de este material, fue necesario contar con un sistema computacional de manejo de base de datos capaz de automatizar las tareas de búsqueda, ubicándolos de manera física los soportes requeridos, manejando la gran cantidad de información que se tiene de manera precisa y clara, proporcionar seguridad a esta, haciéndola confiable, de manera automatizada.

En base a lo mencionado anteriormente se pensó en el Sistema de Automatización de la Fonoteca Alejandro Gómez Arias (SFAGA), que se basa en los fundamentos de la ingeniería de software y de bases de datos relacionales, utilizando la infraestructura existente en las instalaciones de Radio UNAM, además de la intervención directa del personal de la fonoteca quienes utilizarán el sistema cotidianamente.

La programación de SFAGA, demandó el conocimiento de técnicas del modelado de bases de datos, programación en un lenguaje visual, recopilación de información, conocimientos de redes computacionales, capacidad para la solución de requerimientos que surgían durante el diseño, desarrollo e implementación, teniendo una comunicación continua con el usuario final.

Se puede concluir finalmente que la fonoteca “Alejandro Gómez Arias” cuenta con un sistema computacional que les permite resolver sus necesidades y problemática, proporcionando la automatización de las tareas hechas por los responsables de este departamento.

APENDICE A

GLOSARIO

GLOSARIO

API. (Application Programming Interface, Interfaz de Programación de Aplicaciones). Una API es una serie de funciones que los programas de aplicaciones usan para hacer que el sistema operativo haga el trabajo. Los programadores pueden, por ejemplo, abrir archivos y ventanas, mostrar mensajes y hacer otras cosas - desde lo simple a lo complicado - con sólo tener una línea de código que "llama" a la API necesaria.

BASIC. Familia de lenguajes de programación. Fue originalmente ideado como una herramienta de enseñanza, se diseminó entre las microcomputadoras caseras en la década de 1980, y sigue siendo popular hoy en día en muchos dialectos bastante distintos del original. El nombre de BASIC, significa Beginner's All-purpose Symbolic Instruction Code1 (Código de Instrucción Simbólico Todo Propósito para Principiantes), está ligado al nombre de un trabajo sin publicar del coinventor del lenguaje, Thomas Kurtz (el nombre no está relacionado con la serie de C.K. Ogden, “Basic English”).

Browser. Aplicación que permite obtener información de Internet, navegador de Internet.

Cluster. Un cluster es un tipo particular de computadora paralela, es decir, un conjunto de computadoras que pueden trabajar de manera coordinada en la solución de un mismo problema.

Codificar. Hacer o formar un cuerpo de leyes metódico y sistemático. Transformar mediante las reglas de un código la formulación de un mensaje.

DirectX. Es un tipo de API llamado Hardware Abstraction Layer, (HAL) que actúa entre Windows y varios tipos de hardware. El estándard DirectX incluye Direct3D (que acelera las gráficas tridimensionales), DirectSound (para tarjetas de sonido), DirectDraw (para gráficas vectoriales), DirectVideo (para archivos AVI y otros tipos de video), DirectPlay y DirectInput.

FAT32. La palabra FAT, que viene del inglés "File Allocation Table", es el espacio del disco duro destinado a guardar las direcciones de los clusters donde se guardan nuestros ficheros, tanto programas como archivos, osea todo cuanto hay en el disco duro, para que sea fácil su localización, algo así como el índice alfabético de nuestra agenda de teléfonos.

Fonoteca. Colección o archivo de cintas o alambres magnetofónicos, discos, etc., impresionados con la palabra hablada, con música u otros sonidos.

GLOSARIO

Hardware. Conjunto de los componentes que integran la parte material de una computadora.

HTML. Es un lenguaje de programación que se utiliza para la creación de páginas en la WWW. Por página se entiende el documento que aparece en el visualizador. Se compone de una serie de comandos, que son interpretados por el visualizador, o programa que se utiliza para navegar por él.

WWW. En última instancia es el visualizador el que ejecuta todas las órdenes contenidas en el código HTML, de forma que un visualizador puede estar capacitado para unas prestaciones, pero no para otras

Interfaz. Conexión física y funcional entre dos aparatos o sistemas independientes.

MySQL. Es un sistema de administración para bases de datos relacionales (rdbms) que provee una solución robusta a los usuarios con poderosas herramienteas multi-usuario, solucines de base de datos SQL (structured Query Language) multi-threaded. Es rápido, robusto y facil de utilizar.

NTFS. (siglas en inglés de New Technology File System). Es un sistema de archivos diseñado específicamente para Windows NT, con el objetivo de crear un sistema de archivos eficiente, robusto y con seguridad incorporada desde su base. También soporta compresión nativa de ficheros y encriptación (esto último sólo a partir de Windows 2000). NTFS permite definir el tamaño del cluster, a partir de 512 bytes (tamaño mínimo de un sector) de forma independiente al tamaño de la partición. Es un sistema adecuado para las particiones de gran tamaño requeridas en estaciones de trabajo de alto rendimiento y servidores. Puede manejar discos de hasta 2 Terabytes.

ODBC. (Open DataBase Connectivity). Estándar de acceso a Bases de Datos desarrollado por Microsoft Corporation, el objetivo de ODBC es hacer posible el acceder a cualquier dato de cualquier aplicación, sin importar qué Sistema Gestor de Bases de Datos (DBMS por sus siglas en Ingles) almacene los datos.

OLAP. Acrónimo en inglés de procesamiento analítico en línea (online analytical processing). Es una solución que suministra respuestas rápidas a consultas a una base de datos complejas.

OLE. Object Linking and Embedding Vinculación e Inserción de Objetos.

OpenGL. Es un estandar sobre gráficos por computadora. Hoy día es uno de los estandares gráficos

más conocido del mundo. Desde el punto de vista del programador OpenGL es una API para interactuar con dispositivos gráficos y aceleradoras 3D. Contiene cerca de 150 comandos que ayudan a definir objetos, aplicar transformaciones a esos objetos, cambiar sus propiedades (color, textura, luz...), posición de la cámara... entre otros. Tambien hay que tener claro que OpenGL es una librería gráfica, no posee funciones para el control de Audio, Red o Control de Entrada.

Perl. (Practical Extraction and Report Language). Lenguaje de programación desarrollado por Larry Wall (lwall at netlabs.com) inspirado en otras herramientas de UNIX como son: sed, grep, awk, c-shell, para la administración de tareas propias de sistemas UNIX.

PHP. Lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor.

RDF. (Resource Description Framework). Especificación de un modelo de metadatos, (normalmente implementado como una aplicación de XML) que ha sido desarrollada por el World Wide Web Consortium (W3C).

RSS. Acrónimo que tiene diferentes significados, pero el más aceptado es Really Simple Syndication (sindicación verdaderamente sencilla). Es un formato XML indicado especialmente para sitios de noticias que cambien con relativa frecuencia, cuyos documentos están estructurados en canales que a su vez se componen de artículos.

SFAGA. Sistema Fonoteca Alejandro Gómez Arias

Software. Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

XML. Extensible Markup Language. Es un sistema para definir lenguajes especializados que se usan para transmitir datos formateados. XML está relacionada conceptualmente con HTML. XML es un meta-lenguaje en realidad, un lenguaje usado para crear otros lenguajes especializados.

APENDICE B

MANUAL DE USUARIO

En esta sección se presentarán las pantallas del sistema, los objetivos que persigue y la forma de manejar cada una para el mayor aprovechamiento de la funcionalidad de este sistema.

Pantalla de Bienvenida

Esta pantalla se presentará cada vez que se ejecute el sistema, permite entrar a él como un administrador, el cual tiene permisos para agregar o modificar datos de la base, o como usuario de consulta (figura 1)



Figura 1 Pantalla Bienvenida

Pantalla Usuario Consulta

Si entro al sistema como usuario de consulta se desplegara una pantalla que contiene dos pestañas, la primera permite realizar búsquedas de series (figura 2), permitiendo realizar las búsquedas por nombre de la serie, vigencia, productor, clasificación, coproductor y país (figura 3), o desplegar la lista de todas las series existentes en el acervo de la fonoteca.

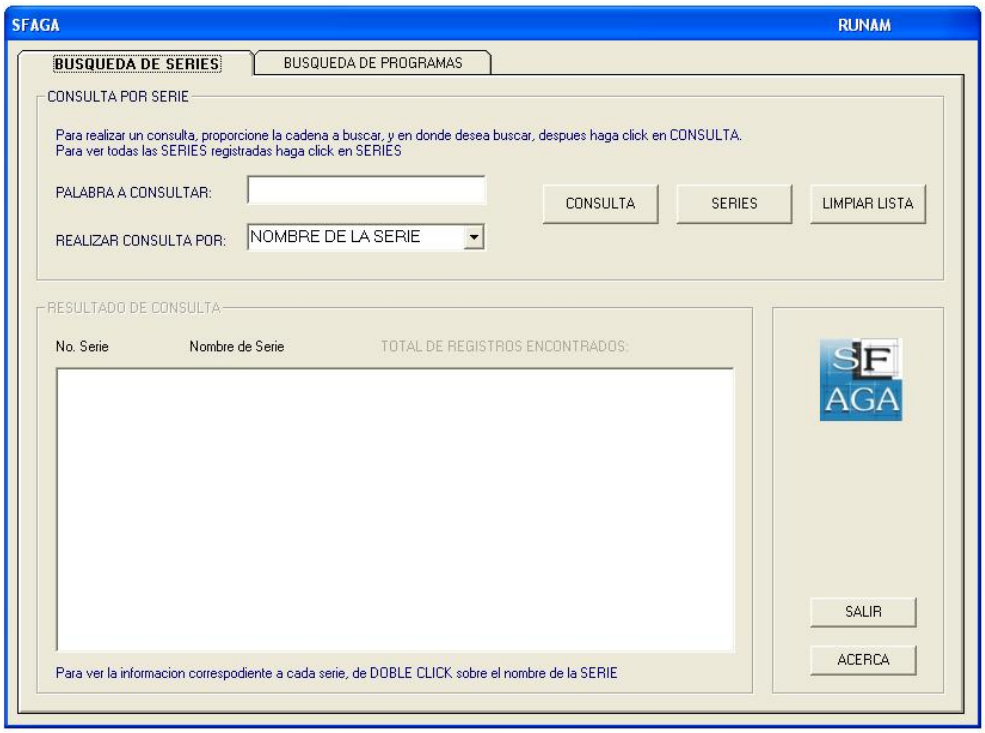


Figura 2 Búsqueda por Series

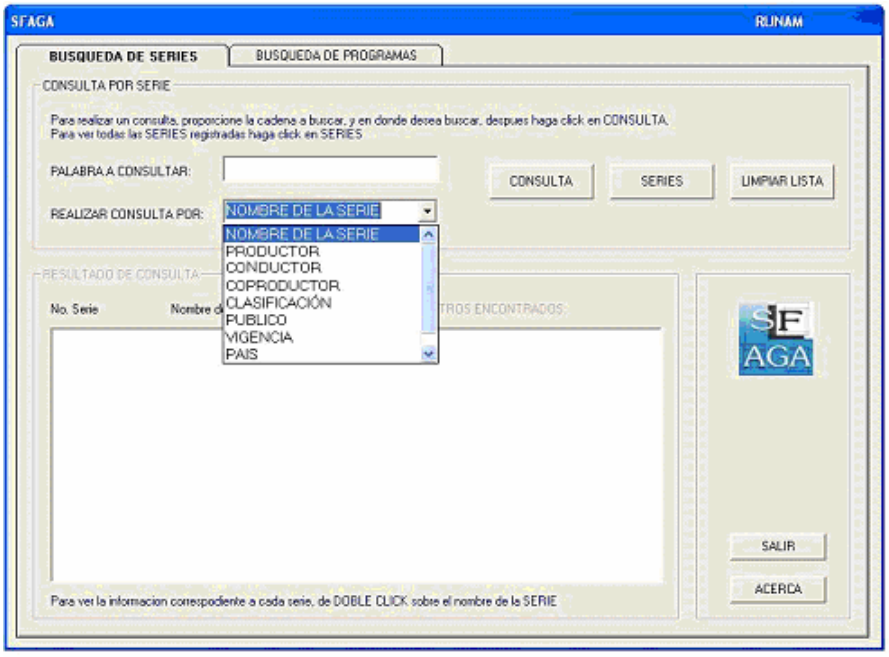


Figura 3 Tipos de Búsqueda de Series

También se encuentran tres botones más, el primero “Limpiar Lista” no permite dejar nuestra lista en blanco para realizar una nueva búsqueda, el siguiente, “Salir”, no permite salir de la aplicación, desplegando un mensaje preguntado si se está seguro de salir del programa (figura 4), el último botón, “Acerca”, muestra otra pantalla en donde se encuentran los datos del sistema, como nombre, versión, los nombres de los realizadores y el nombre de la institución a la que pertenece (figura 5).



Figura 4 Confirmación de Salir



Figura 5 Acerca del Sistema

Una vez que se realizo la búsqueda de la serie deseada y se desplegó el nombre en la lista, se le da doble click sobre la serie para que despliegue sus características en otra pantalla (figura 6).

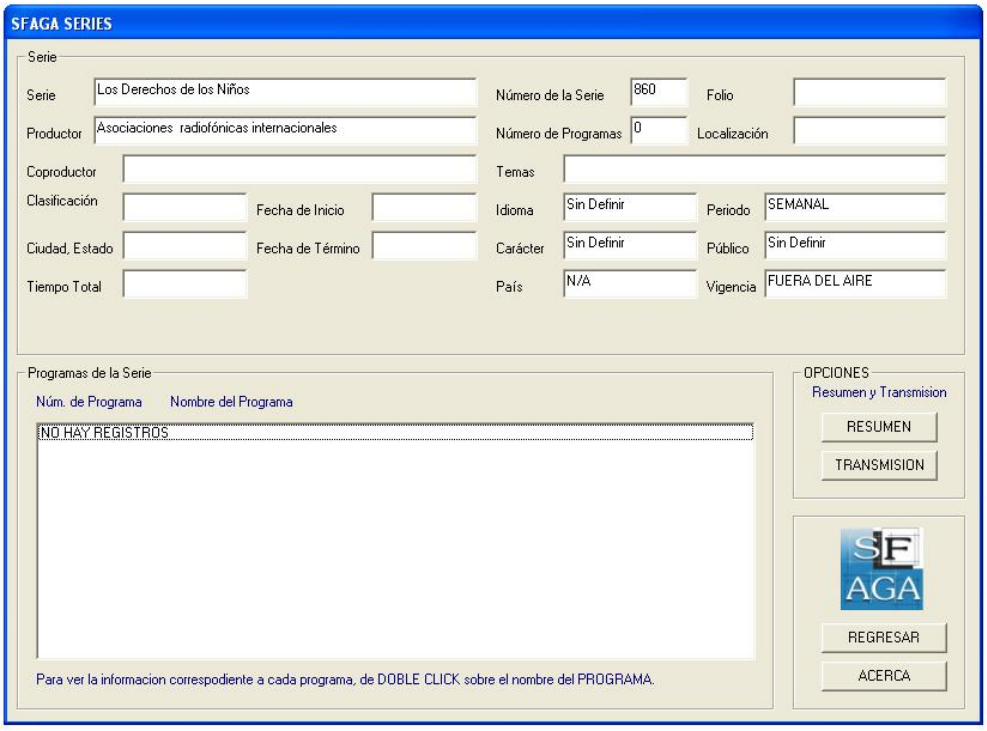


Figura 6. Datos de la serie.

En esta pantalla además de ver los datos generales de cada serie tiene cuatro botones, el primero de ellos despliega el resumen de la serie (figura 7),



Figura 7 Resumen de la serie

el siguiente botón muestra la fecha y la hora de la primera transmisión de la serie, así como las posibles retransmisiones que tenga la serie (figura 8),



Figura 8 Transmisión de la serie

los siguientes botones son para regresar a la pantalla anterior y realizar una nueva búsqueda y para ver la información acaezca del sistema.

En la pantalla donde se despliega la información de la serie (figura 6) también se muestra una lista de los programas pertenecientes a esta serie, al darle doble click al nombre del programa que se eligió desplegara la información de dicho programa en otro pantalla (figura 9).

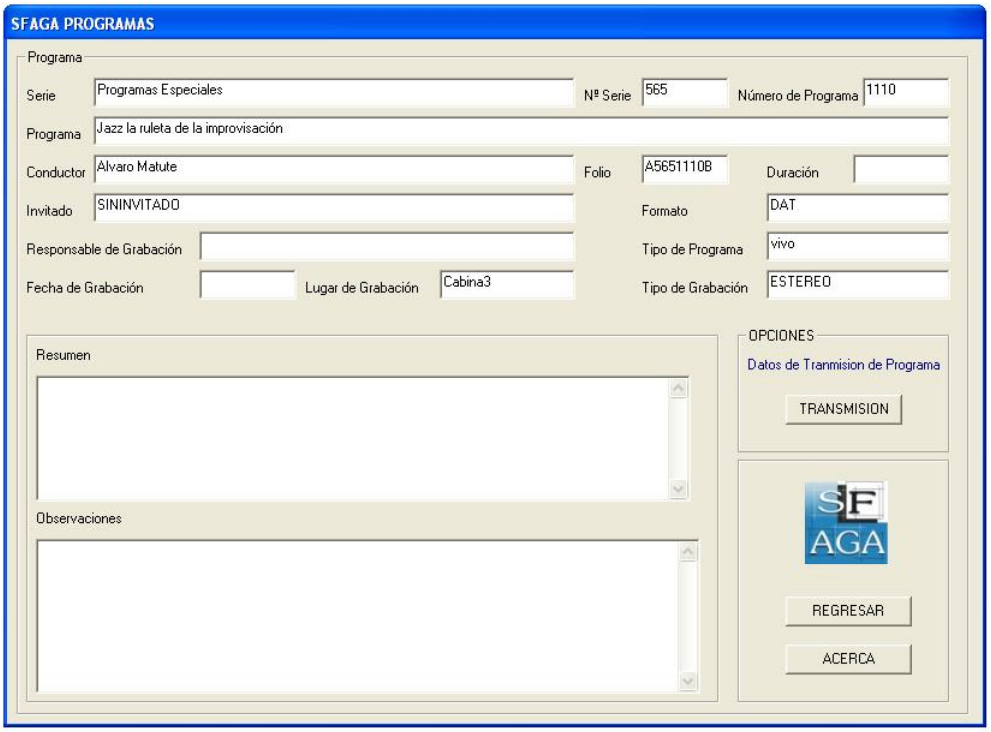


Figura 9 Datos del Programa

Esta pantalla muestra las características del programa, como nombre, a que serie pertenece, así como tres botones, el primero muestra los datos de transmisión y retransmisión del programa (figura 8), los siguientes son los que permiten regresar para realizar una nueva búsqueda y mostrar datos referentes al sistema.

En la pantalla de consulta para usuarios también se pueden realizar búsquedas de programas, esto se muestra en la figura 10, la cual tiene dos botones similares a la pantalla de búsqueda de series, “Salir” y “Acerca”, las cuales ya se mencionaron anteriormente, como en la pantalla de búsqueda de series se puede elegir una de varias formas de búsqueda; nombre del programa, conductor, invitado, entre otros, una vez elegida la opción de búsqueda se oprime el botón “Consulta”, el cual desplegara en la lista todos los nombres de programas que relacionados con el parámetro de búsqueda que se dio, una vez que se encontró en la lista el programa deseado, se le da doble click sobre el nombre del programa para que se muestre otra pantalla desplegando los datos correspondientes a dicho programa, si el programa deseado no se desplegó en la lista se utiliza el botón “Limpiar Lista” para limpiar la lista y se permita realizar

otra búsqueda.

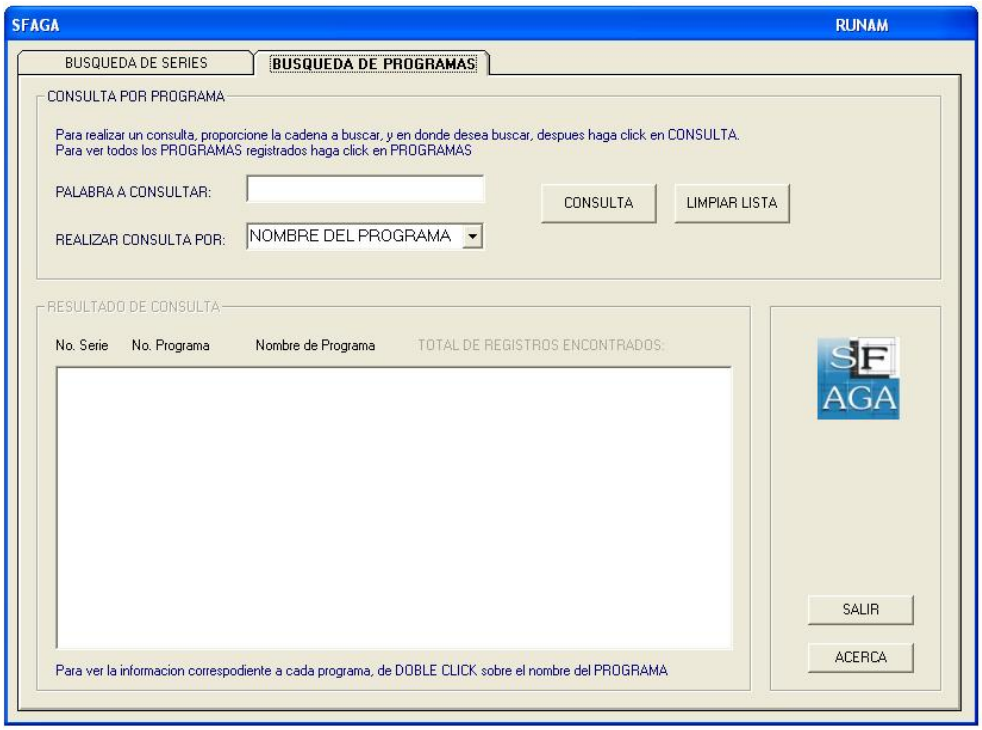


Figura 10 Búsqueda por programas

Y una vez que se encontró el programa deseado se le da doble click sobre el nombre para desplegar en la pantalla los datos relacionados con dicho programa (figura 9).

Si la persona que desea iniciar el sistema es un administrador del mismo, las pantallas mostradas cambian, ya que estas nuevas pantallas permiten la administración de la base de datos, en la figura 11 se muestra esta pantalla.

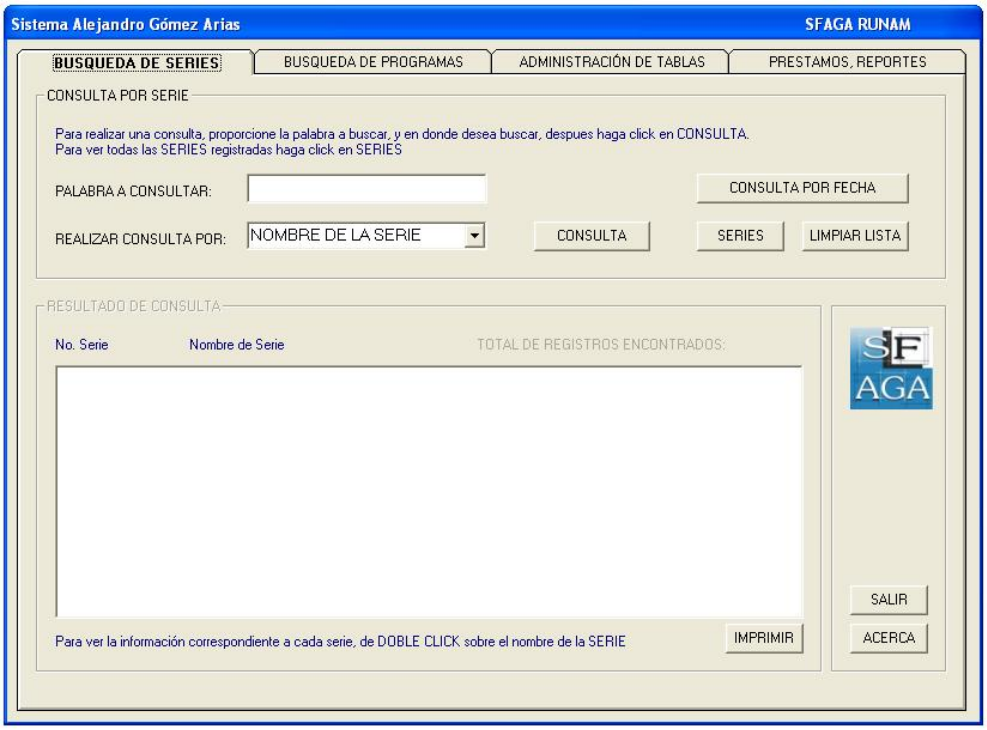


Figura 11 Pantalla administrativa

Esta pantalla contiene cuatros tabs o pestañas, el primero que se muestra es el de búsqueda de series, la cual se maneja de forma perecida a la pantalla que se despliega a los usuarios no administradores, agregando dos botones más, el primero que permite realizar búsquedas por fechas (figura 12) con el formato dd/mm/aaaa, aceptando también espacios en blanco en vez de /, el cual despliega dos opciones de búsqueda, un fecha exacta o un rango de fechas, desplegando la información en la lista de abajo, y otro botón que imprime los resultados de las búsquedas.

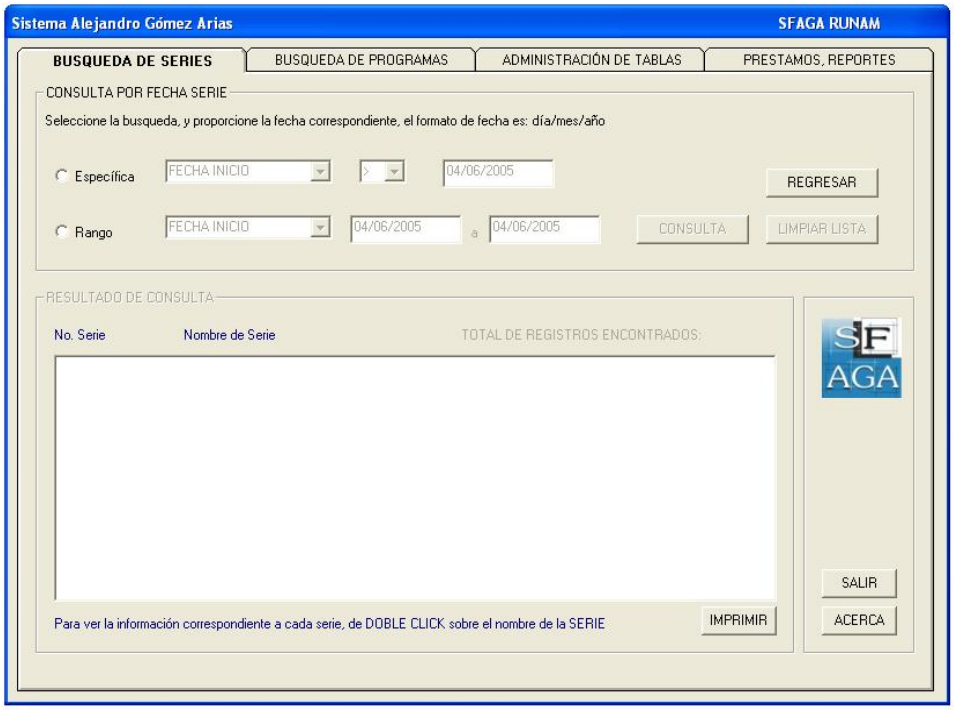


Figura 12 Búsqueda por fechas

Al dar doble click sobre el nombre de una serie desplegada en la lista se presentara la información correspondiente a dicha serie en una pantalla como la mostrada en la figura 13.

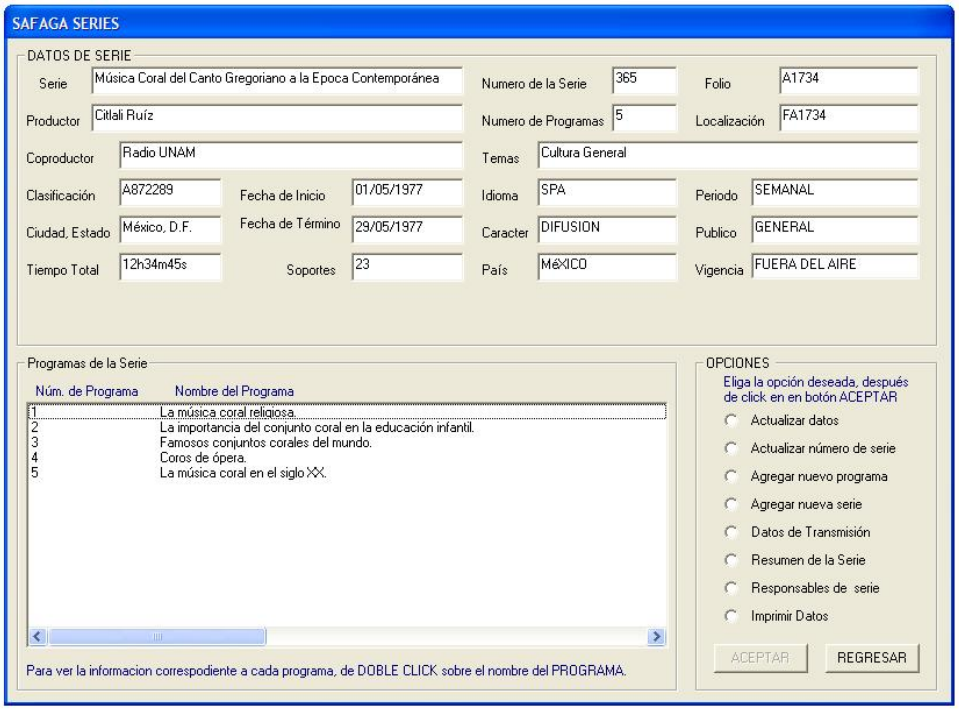


Figura 13 Vista serie administrativa

En esta pantalla además de desplegar la información de la serie y los programas pertenecientes a dicha serie, se despliegan opciones administrativas, como: Actualizar Datos, Actualizar número de serie, Agregar nuevo programa, Agregar nueva serie, Datos de transmisión, Resumen de la serie, Responsable de la serie e Imprimir datos, además de dos botones, “Aceptar” y “Regresar”.

El botón Regresar oculta esta ventana y muestra la ventana de búsqueda administrativa, el botón actualizar varia dependiendo de la opción seleccionada arriba, la opción Actualizar datos oculta las opciones y despliega dos botones, como se muestra en la figura 14, “Actualizar” y “Cancelar”, además permite la escritura en la mayoría de los campos mostrados en la forma, con excepción de el número de serie.

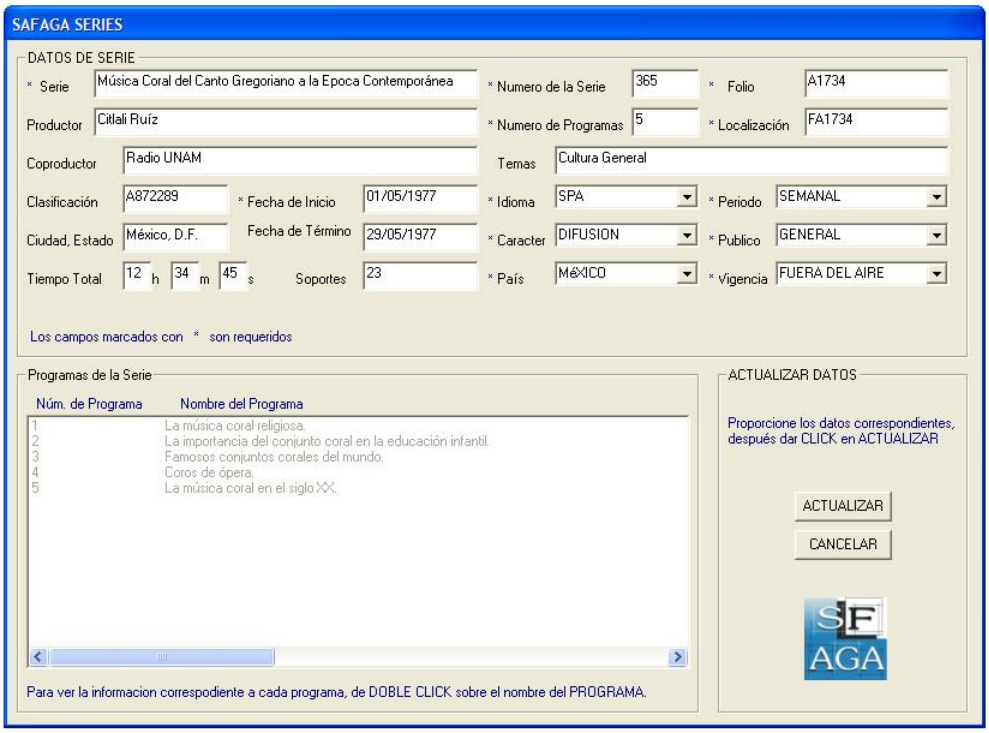


Figura 14 Actualización de datos

El botón “Cancelar” devuelve las opciones de actualización e impresión, y si se realizo un cambio en los datos los ignora dejando los datos originales, el botón actualizar guarda en la base de datos los valores capturados en esta pantalla, validando antes que los campos

requeridos tengan información coherente, como en el caso de la fecha, además en los campos de este tipo solo acepta el formato antes mencionado para las fechas, y para insertar el folio la caja de texto solo acepta “A” o “B” al principio seguido por cualquier número. Si los datos son correctos se despliega el mensaje de que se actualizaran los datos, figura 15, con dos botones, el primero que es para “Aceptar” la inserción de los datos a la base, y el botón “Cancelar”.

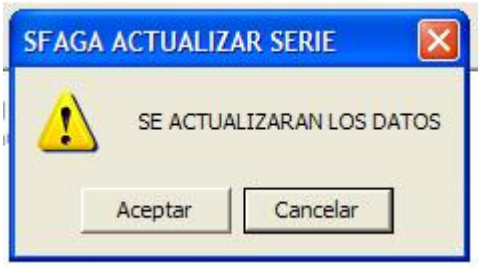


Figura 15 Actualizar Serie

Si existen errores en el formato de los campos requeridos manda mensajes de advertencia, figura 16.

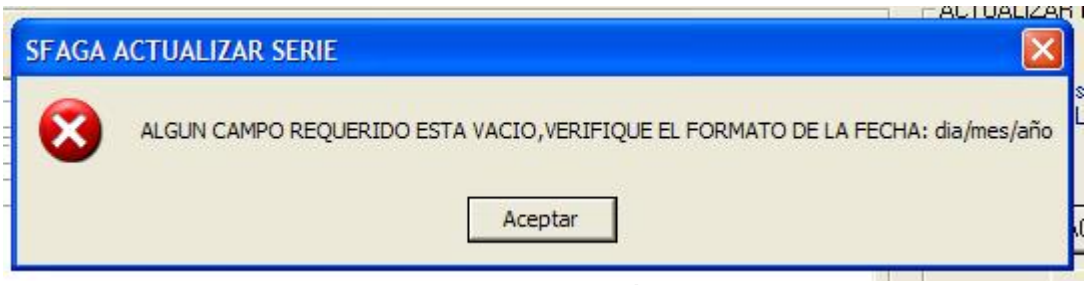


Figura 16 Datos erróneos

Al terminal la actualización de la serie regresa las opciones de actualización e impresión que se muestran en la figura 14.

La siguiente opción que se muestra en el listado es la de “Actualizar número de serie” la cual cambia de color el campo donde esta el número de la serie y lo habilita para poder escribir en él, además de mostrar dos botones, “Aceptar” que manda el número una vez modificado y “Cancelar” que deshabilita la escritura en el campo y lo regresa a su color anterior y regresa las opciones de actualización e impresión, figura 17.

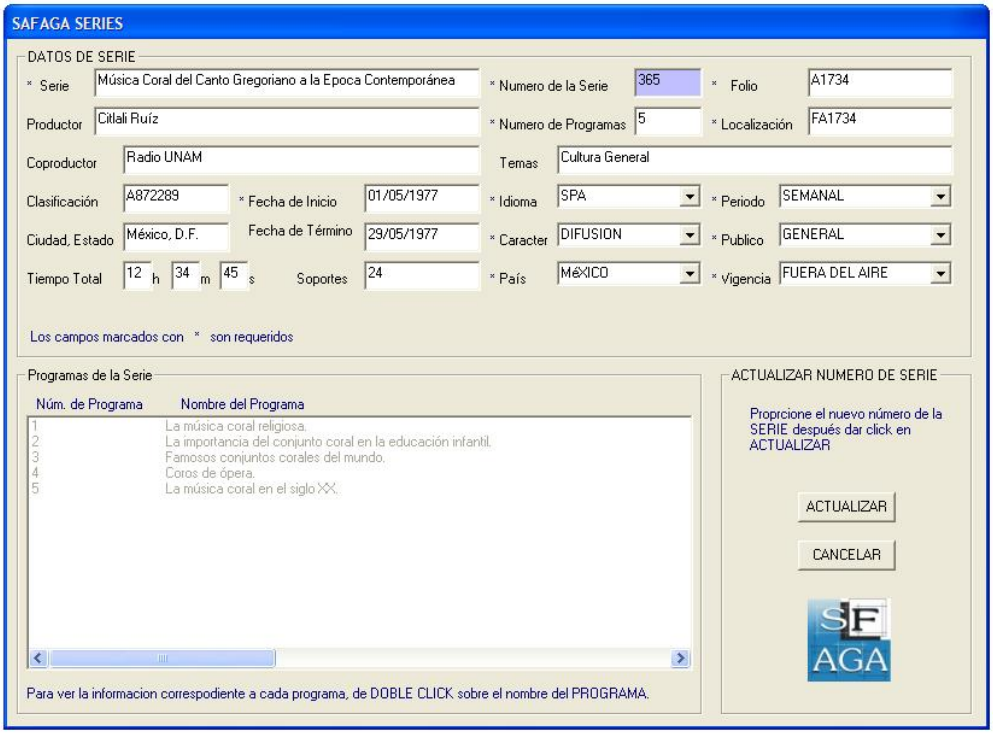


Figura 17 Actualizar número de serie

Si al actualizar el número de la serie ya existe se desplegara un mensaje de advertencia, figura 18.



Figura 18 Serie existente

Si no es el caso se desplegara un mensaje que pedirá si se desea continuar con la actualización o no.

La siguiente opción es “Insertar nuevo programa” la cual despliega la pantalla donde se solicitan los datos para insertar un nuevo programa a la base de datos, perteneciente a la serie de la cual se invoco esta acción, figura 19.

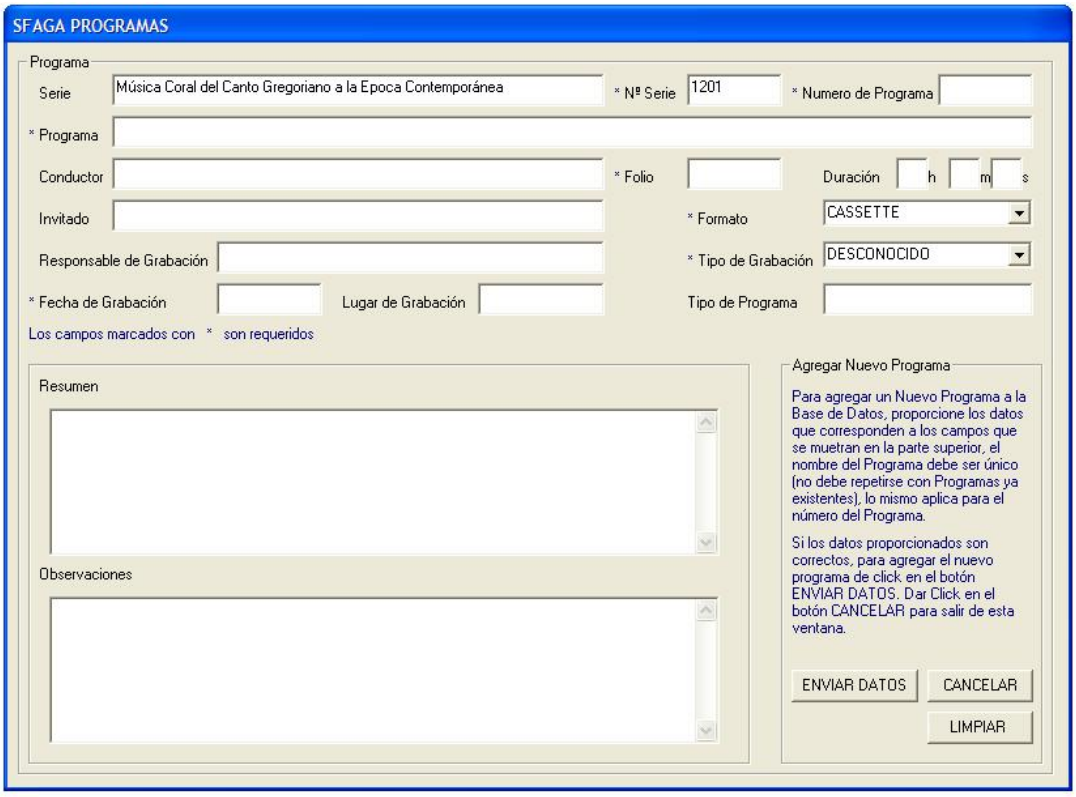


Figura 19 Insertar nuevo programa

Los botones de esta pantalla permiten insertar los datos a la base de datos (Enviar Datos), el cual verifica que los datos mínimos requeridos sean capturados (figura 20) y correctos (figura 21), como el caso de las fechas y el folio, manejando el mismo criterio antes mencionado, “Cancelar” regresa a la pantalla de la serie y el botó “Limpiar” limpia los campos de la pantalla.



Figura 20 Datos requeridos capturados



Figura 21 Datos requeridos validos

Si los datos son correctos se manda un mensaje en el que se especifica a que serie pertenece el programa que se acaba de crear (figura 22).



Figura 22 Serie a la que pertenece el programa

La siguiente opción es “Insertar nueva serie” la cual vacía y habilita todos los campos de la pantalla, desplegando tres botones, el primero envía la información la información capturada de la nueva serie a la base de datos, el segundo cancela la acción y el botón “Limpiar” deja todos los datos vacíos para insertar datos nuevos. Figura 23



Figura 23 Insertar nueva serie

La siguiente opción permite ver los datos de transmisión de la serie, además de incluir un botón que permite la inserción o actualización de los datos de la serie (figura 24).

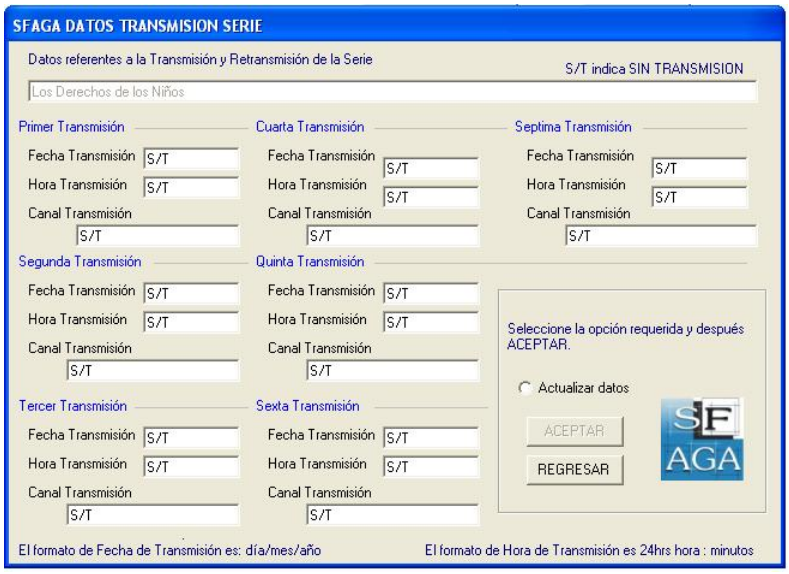


Figura 24 Transmisión de Serie administrativa

La siguiente opción despliega el resumen de la serie, con la opción de actualizarlo (figura 25).

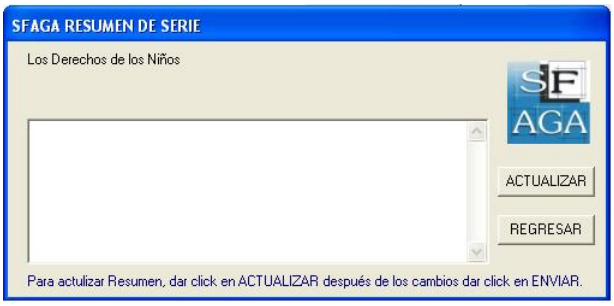


Figura 25 Resumen de la serie administrativo

La opción “Responsable de la serie” despliega una pantalla en la que se muestran los datos de la persona encargada de insertar la serie en la base de datos y la ultima persona que realizo cambios a los datos. (Figura 26).



Figura 26 Responsable de la serie

La opción “Imprimir datos de serie“ imprime en un formato establecido los datos completos de la serie (figura 27), y la opción “Imprimir programas de la serie” imprime todos el nombre de los programas que pertenecen a la serie (figura 28).



Figura 27 Imprimir serie



Figura 28 Imprimir programas de la serie

Al entrar como administrador también se pueden realizar búsquedas por programas, como cuando se entra como un usuario normal, con la diferencia de que permite realizar además búsquedas por fechas y la impresión de la lista de los resultados de las búsquedas. (Figura 29).

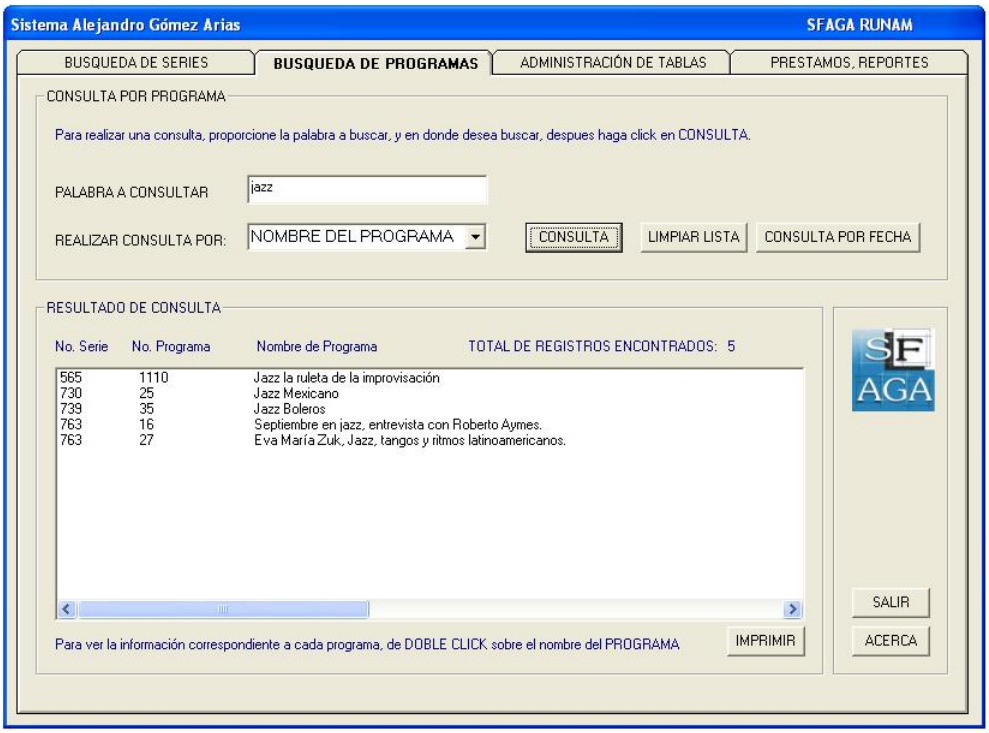


Figura 29 Búsqueda administrativa de programas

Donde los botones que se muestran son el de “Consulta” que dependiendo de lo elegido en el combo y la palabra a buscar despliega la lista de las coincidencias, el botón “Limpiar lista” limpia los resultados obtenidos anteriormente en la lista, “Consulta por fecha” despliega nuevas cajas de texto don de se selecciona una fecha fija a un rango de fechas para realizar la búsqueda (figura 30).

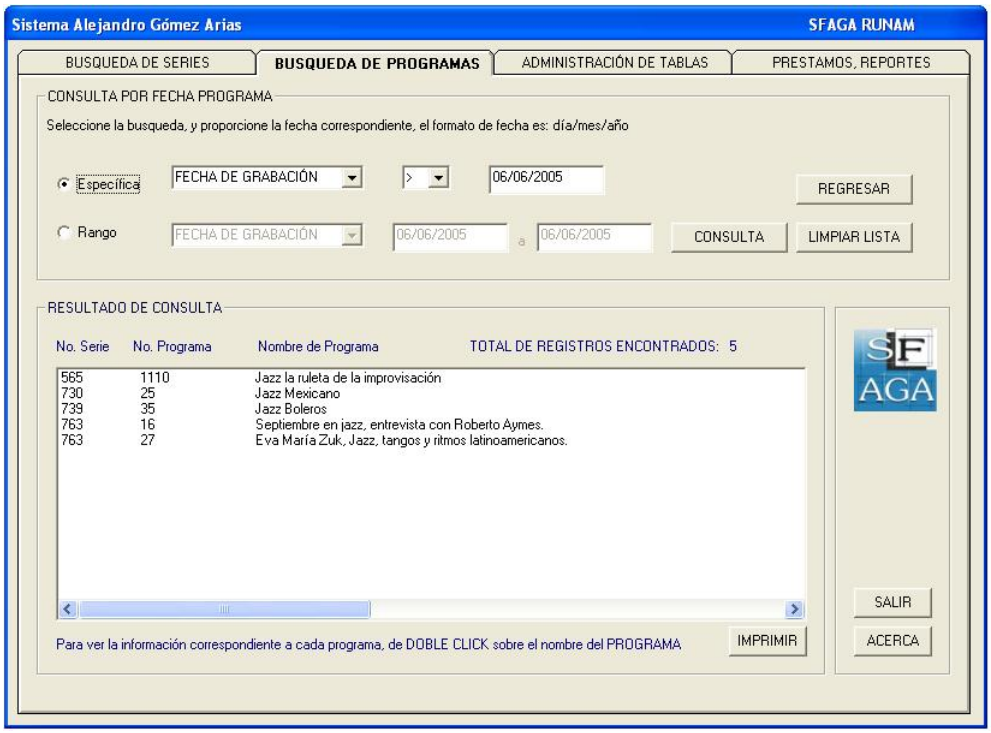


Figura 30 Búsqueda de programas por fechas

El botón imprimir, permite la impresión de los resultados de la búsqueda.

Una vez que se encontró el programa deseado se da doble click sobre el nombre para que muestre la información de dicho programa en una pantalla nueva como se ve en la figura 31, a esta pantalla también se llega al estar en los datos de la serie (figura 13), donde se muestra los programas pertenecientes a ella, dando doble click sobre uno de ellos.

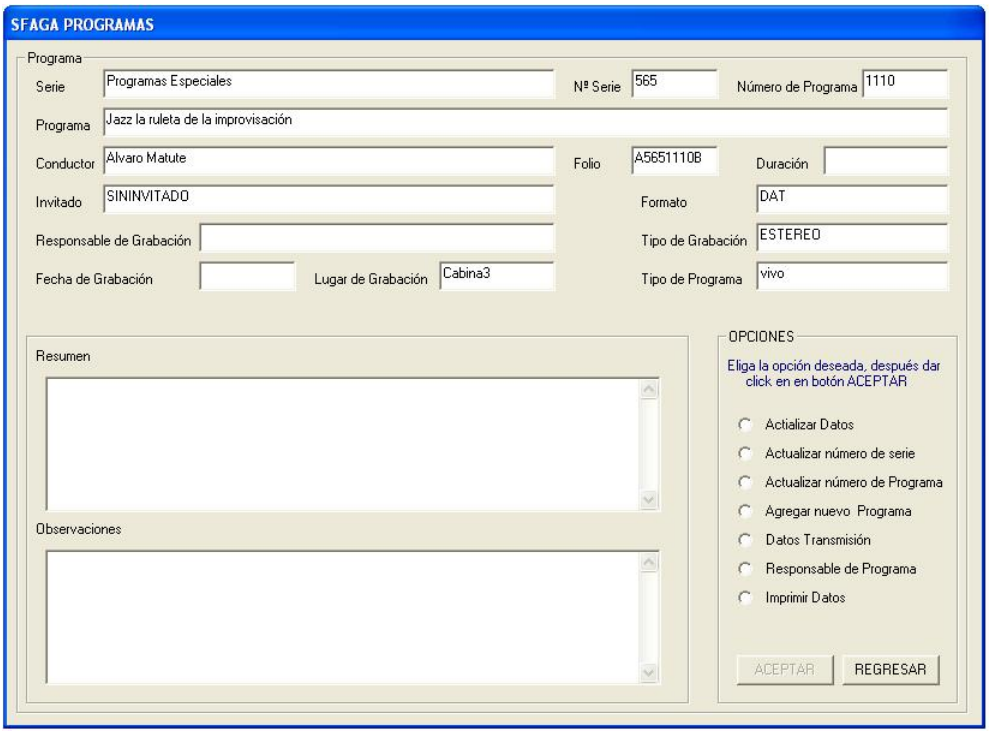


Figura 31 Administración del programa

En esta pantalla además de ver los datos del programa seleccionado como son el nombre, la serie a la que pertenecen, el folio el número de la serie, entre otros, también muestra opciones para la actualización de este programa en particular, la primera de ellas es la de “Actualizar Datos” la cual habilita la escritura en las cajas de texto para poder modificar los datos (figura 32), desplegando también dos botones, el primero es el que envía los datos actualizados a la base de datos, el segundo cancela la actualización y desase los cambios que se hayan realizado. Para actualizar se verifica que los datos nuevos tengan el formato correcto, como en el caso de las fechas (dd/mm/aaaa) y el folio que empiece por “A” o “B”, y los campos solo numéricos.

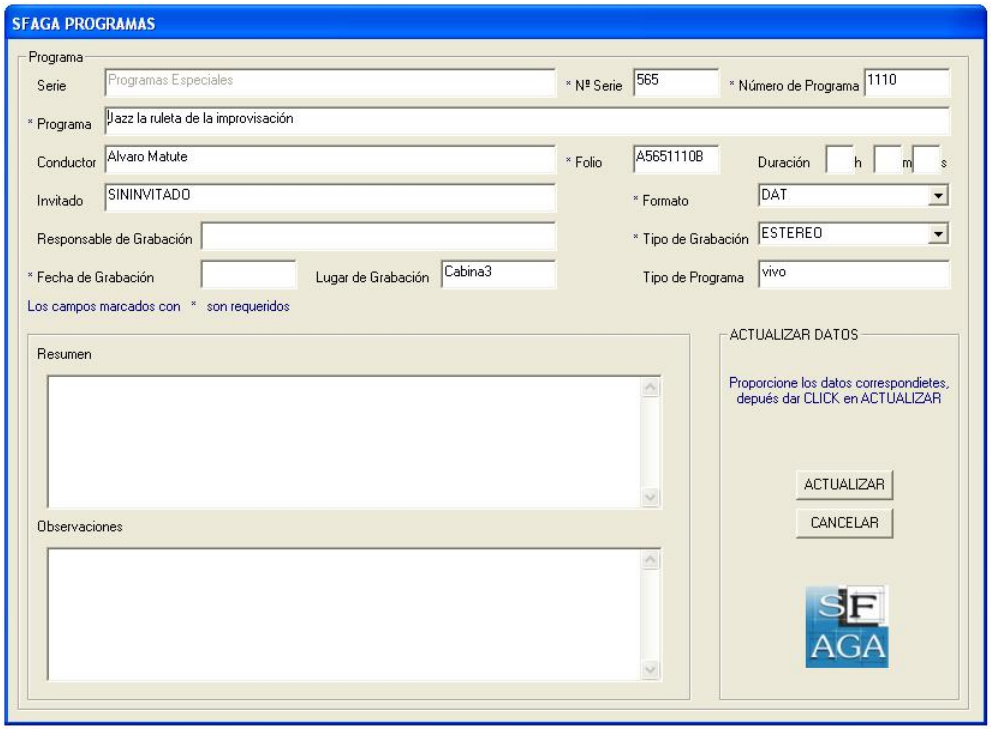


Figura 32 Actualización del programa

La siguiente opción es la de la actualización de la serie, permite cambiar el programa de serie cambiando el número, figura 33, mostrando dos botones, “Actualizar” que manda a la base de datos el número de serie actualizada, y el botón “Cancelar” que regresa al número anterior de la serie.

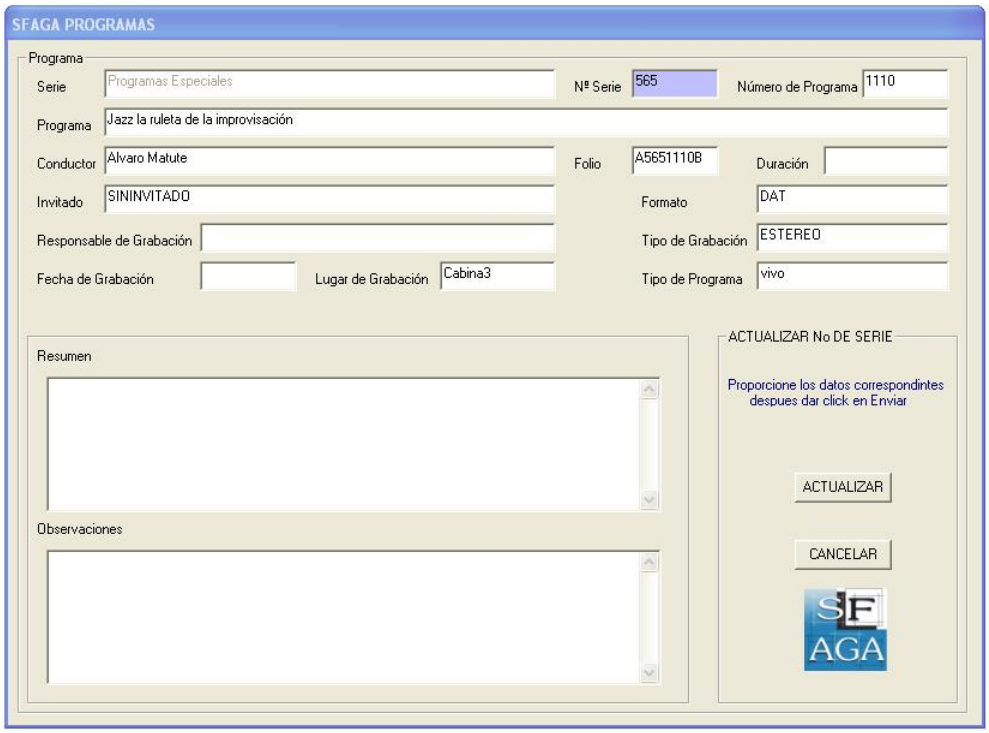


Figura 33 Actualizar número de serie

Si el número de la serie a la que se quiere pasar el programa no existe, el sistema preguntara si se desea agregar la nueva serie, si se da aceptar, despliega la pantalla de insertar nueva serie (figura 23), si elige cancelar, se ignoran los cambios realizados, también se verifica que el número de programa no exista en la serie nueva, si es así se muestra un mensaje de error.

La opción “Actualizar número de programa” permite cambiar el número que identifica al programa dentro de la serie, si el número que se le quiere asignar al programa ya existe manda un mensaje de error, los botones para esta opción son “Actualizar“ que manda el nuevo número a la base de datos y “Cancelar” que elimina los cambios realizados. (Figura 34)

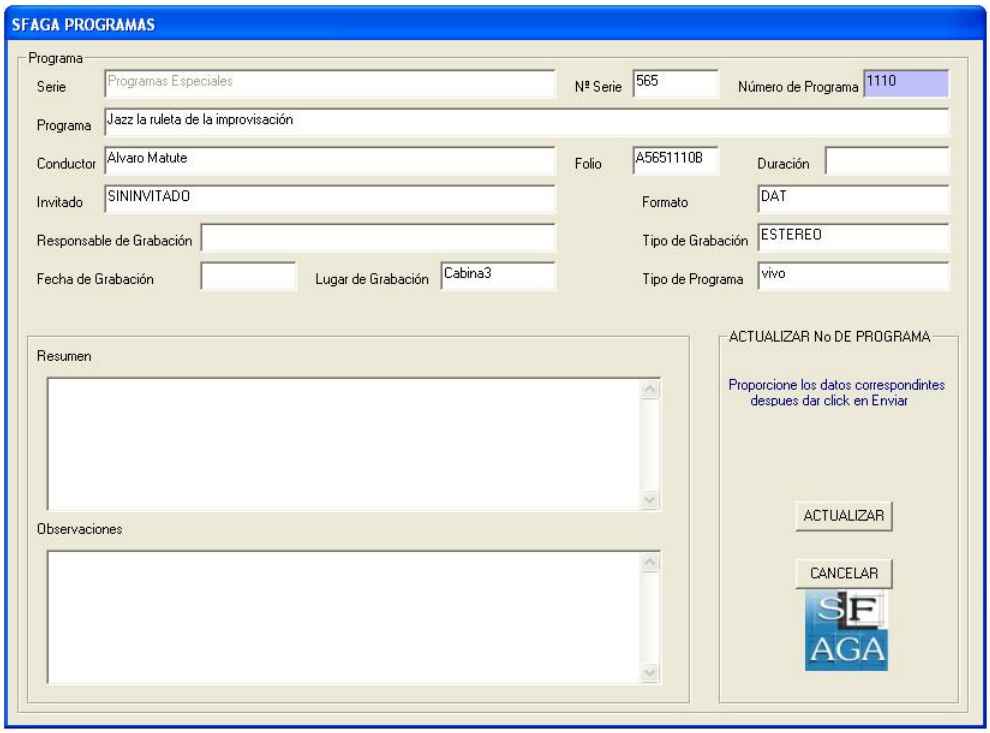


Figura 34 Actualizar número de programa

La siguiente opción agrega un nuevo programa a la base de dato, la pantalla que se despliega en esta opción se muestra en la figura 19.

Los datos de transmisión del programa se muestran en una pantalla nueva, donde se puede actualizar o insertar una nueva fecha y hora de transmisión (figura 24). La opción “Responsable del programa” despliega una pantalla nueva con los datos de la persona responsable de insertar el programa a la base de datos y los datos del responsable de su actualización (figura 26), la última opción muestra las opciones de impresión figura35.

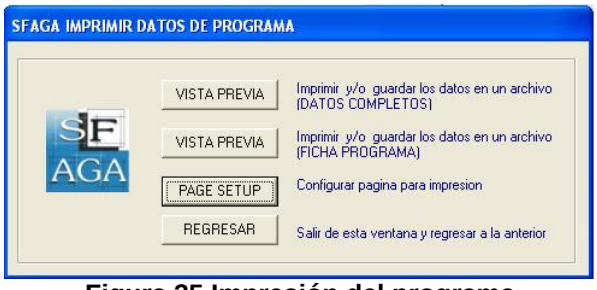


Figura 35 Impresión del programa

El tercer tab o pestaña es la administración de tablas menores, tales como departamentos, responsables de la fonoteca, interesados en obtener préstamos y otras (figura 36).

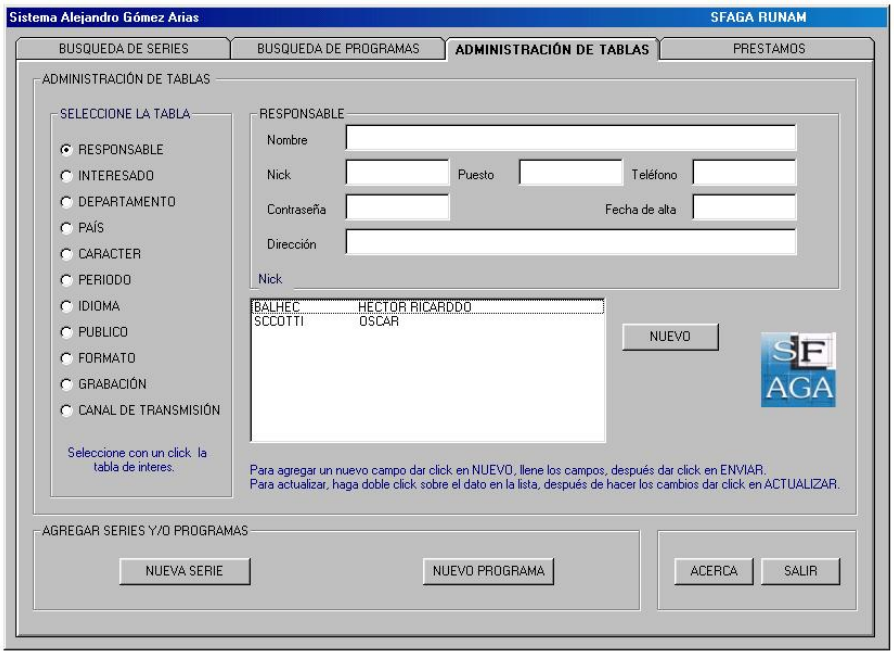


Figura 36 Tabla de Responsables

En esta pantalla se encuentran cuatro secciones, la primera consiste en una lista en donde se nombran las diversas tablas, el primer elemento de la lista es Responsable, la segunda sección muestra una lista de los datos almacenados en la tabla de los responsables de la fonoteca, y campos en donde se puede agregar un nuevo responsable o editar uno existente, para un nuevo responsable se debe presionar el botón que se encuentra junto a la lista (NUEVO), con lo que se permite la inserción de datos en las cajas de texto, si se desea editar un responsable que se encuentra en la base, se debe dar doble click sobre el nombre del responsable en la lista y automáticamente desplegara los datos completos en las cajas de texto de arriba, habilitando la capacidad de insertar y modificar los datos en cada una de ellas.

El segundo componente en la lisita es el de Interesado, la forma de almacenar y modificar datos de todas esta tablas es similar, por lo que no se verán las demás tablas, en forma grafica, se muestran otros botones que ya se han visto anteriormente, “Salir” que termina el sistema,

“Acerca” que muestra datos referentes a la aplicación, y los botones de “Insertar nuevo programa” e “Insertar nueva serie” que muestra las pantallas mostradas en la figura 19 y la figura 23 respectivamente.

La última pestaña o tab es la referente a los préstamos realizados a otras áreas de la radiodifusora y para obtener reportes históricos de interés para las personas de la fonoteca (figura 37).



Figura 37 Préstamos y reportes

La primera parte de esta pantalla hace referencia a los nuevos préstamos, los programas que en este momento están en préstamo o tienen deuda, ver que interesado tiene el material en este momento y realizar un búsqueda exacta de un programa prestado en especifico, para realizar estas tareas se cuentan con campos que permiten seleccionar que se desea hace, para insertar un nuevo préstamo se muestra una pantalla como la que se ve en la figura 38, en donde se selecciona de una lista la personas interesadas en obtener dicho servicio, la fecha del día en que se pidió el préstamo la fecha cuando se entrega el programa, para realizar el

préstamo es necesario contar con el folio del programa, con el cual se otorga el préstamo y genera la clave de préstamo, se pulsa el botón “Crear Préstamo”.

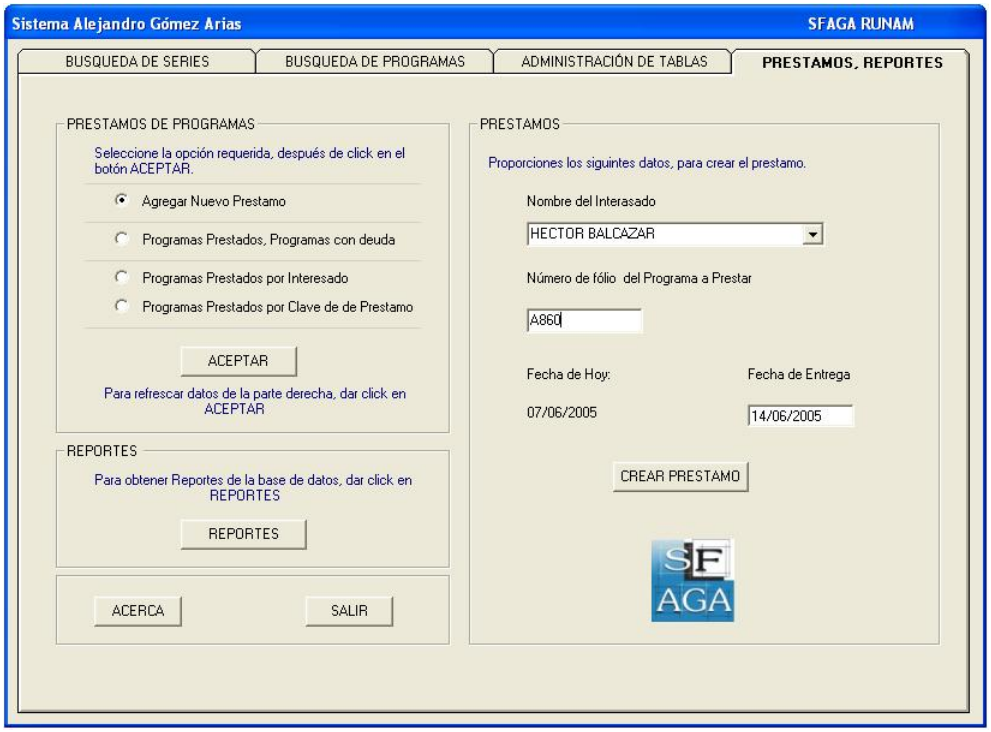


Figura 38 Nuevo préstamo

Si el folio proporcionado es incorrecto el sistema le mandara un mensaje de error donde se especifica que ningún programa tiene el folio proporcionado, si el folio es correcto se devuelve el nombre del programa y se crea el préstamo, pero si el programa ye esta prestado manda un mensaje especificando que el programa esta en préstamo.

En la siguiente opción, “Programas prestados, programas con deuda”, se muestran dos listas, la primera con los nombres de los programas que se encuentran prestados, en la segunda lista se muestran los programas que no han sido devueltos. Figura 39

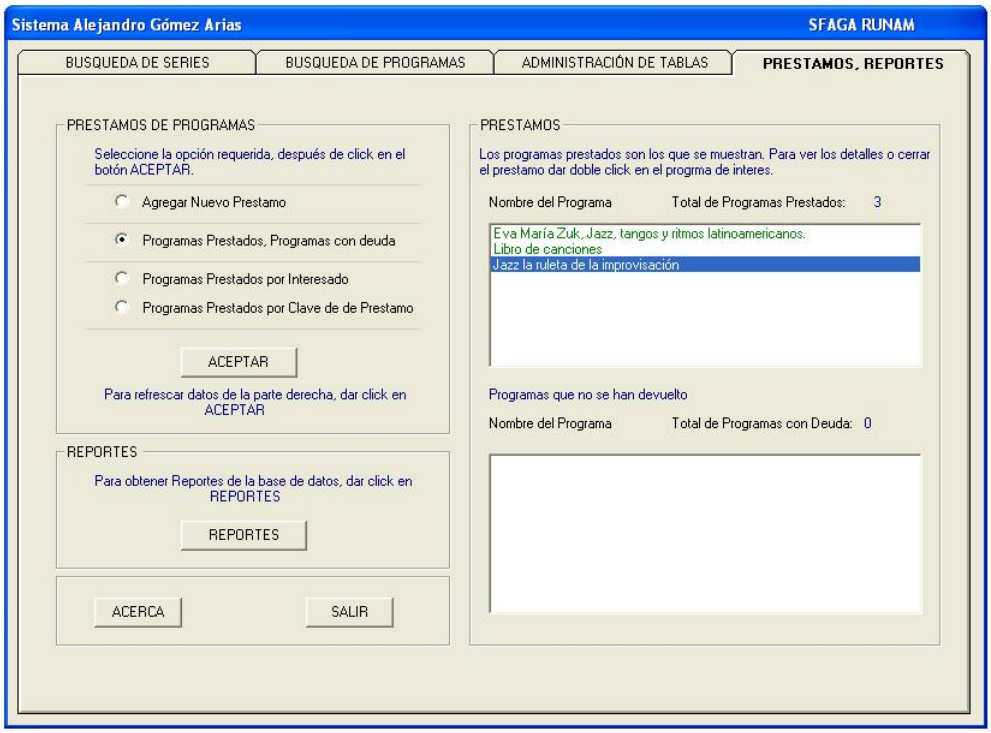


Figura 39 Programas prestados y en deuda

Para ver que algunas de la características del programa, nombre, serie a la que pertenece, responsable de almacenarlo, etc.; y las características del préstamo, como la fecha en que se realizo, la fecha de devolución, se le da doble click al nombre del programa que desplegara otra pantalla con los datos antes mencionados (figura 40). En esta pantalla también se puede cerrar el préstamo un vez devuelto el material (botón “Cerrar Préstamo”).

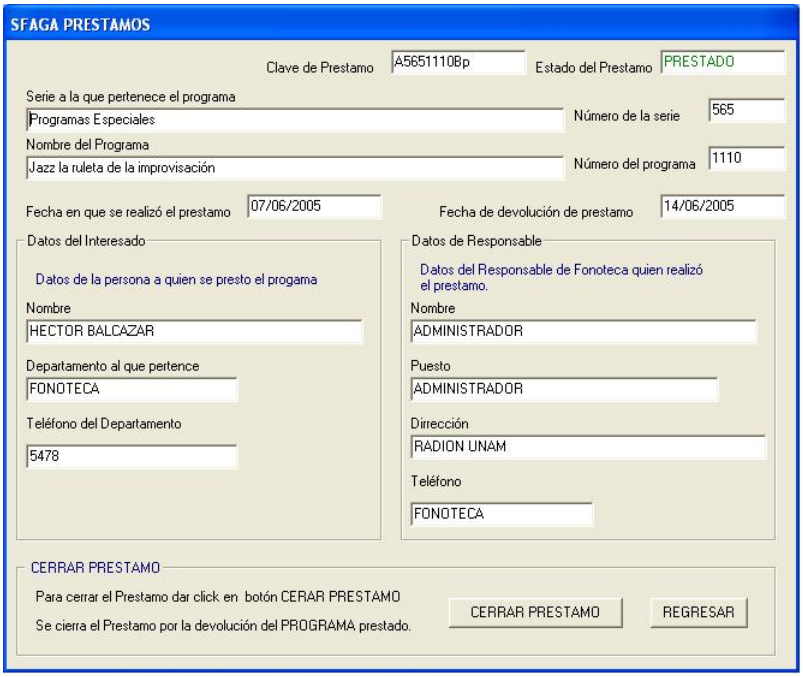


Figura 40 Estado del préstamo

En la siguiente opción se verifica que programas están prestados a que persona de la radiodifusora proporcionando el nombre (figura 41), si se da doble click, una vez que se desplieguen los nombres de los programas que tiene la persona, sobre uno de ellos muestra los datos del préstamo y del responsable en otra pantalla (figura 40).

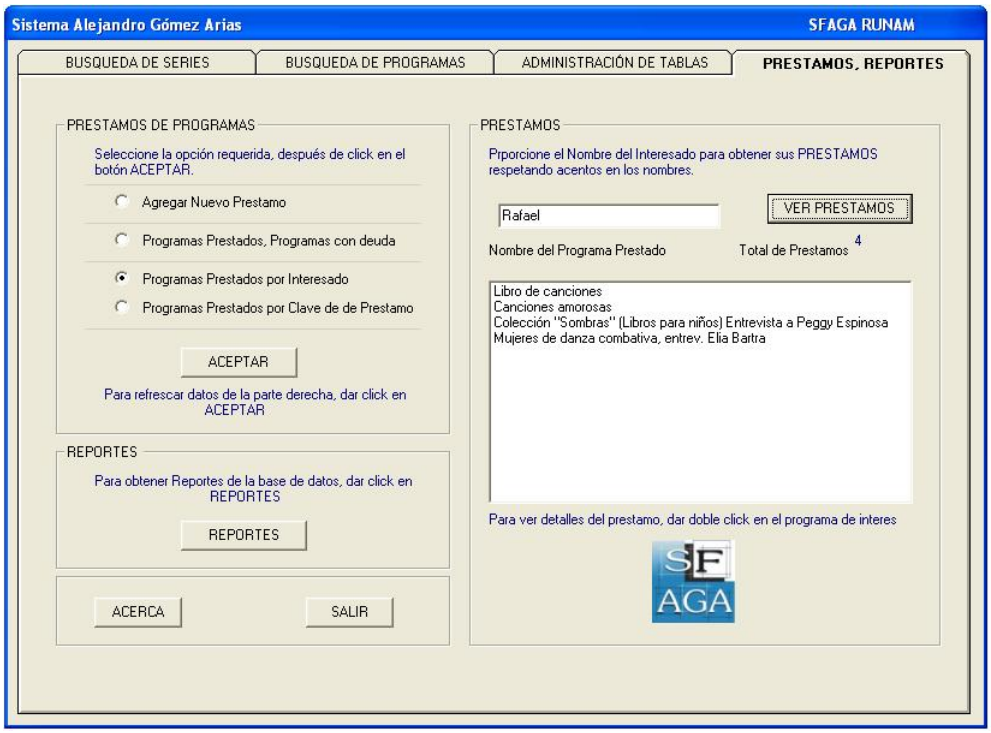


Figura 41 Préstamos por interesado

La última opción en cuanto a préstamos es “Programas prestados por clave”, la cual permite de una manera más rápida ver el estado del préstamo (figura 40), ya que sabiendo la clave del préstamo se encuentra el programa directamente sin tener que consulta las listas antes mencionadas, esta opción se muestra en la figura 42.

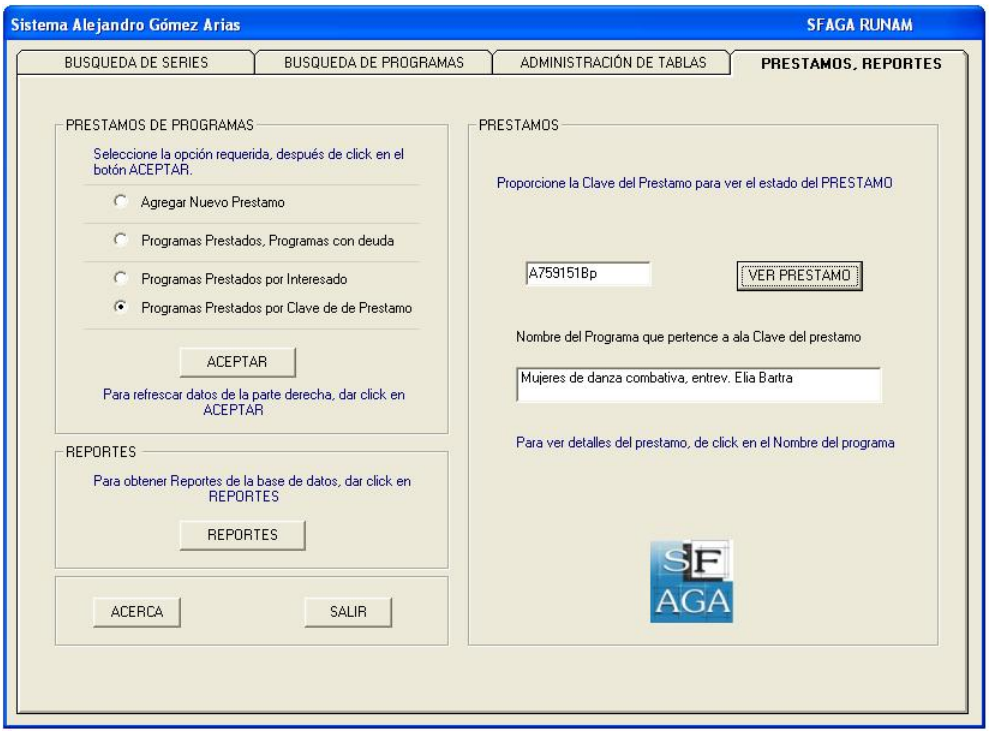


Figura 42. Préstamos por clave

Lo siguiente que se muestra es el botón “Reportes”, el cual despliega varios botones más, los cuales permiten observar diversos reportes que genera el sistema, los primeros son referente a los canales de transmisión y cuales de estos son los mas utilizados, ya se en una vista general, mostrando todos los canales de transmisión y el número de programas que se han transmitido por él, o un reporte mas especifico mencionando el canal de transmisión de interés, figura 43.

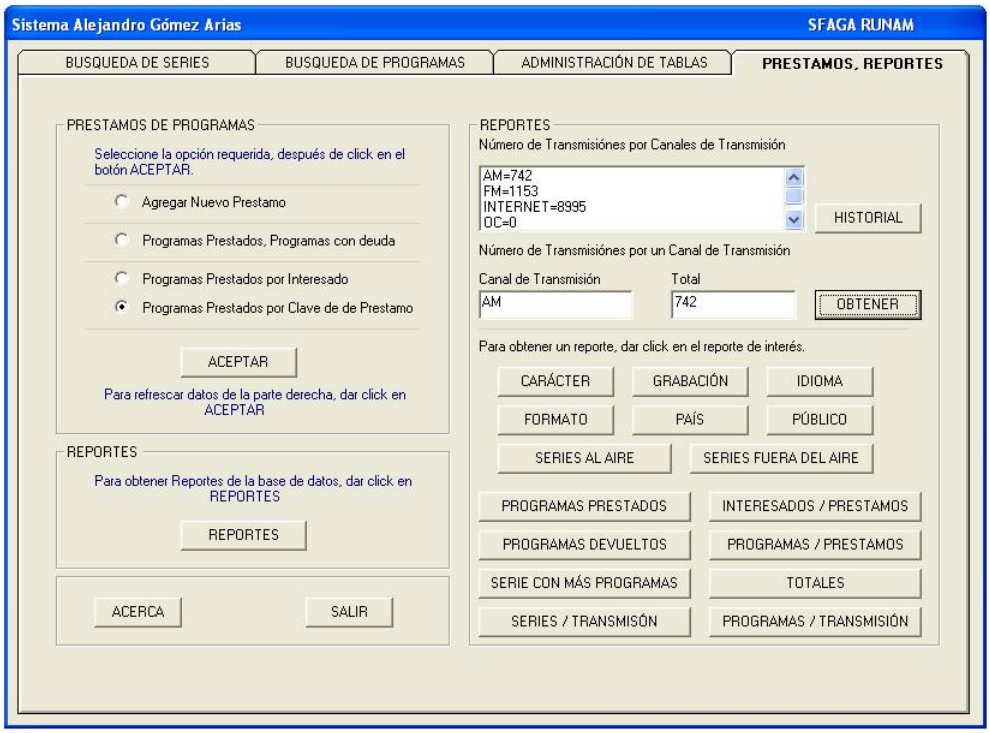


Figura 43. Reportes por canal de transmisión

Los botones de “Carácter”, “Grabación”, “Idioma”, “Formato”, “País” y “Publico”, despliega una pantalla similar, en la cual se muestra el tipo de dato especificado en cada uno, con el número de programas o series que pertenecen a dicha nomenclatura, un ejemplo de estas pantallas se ve en la figura 44.

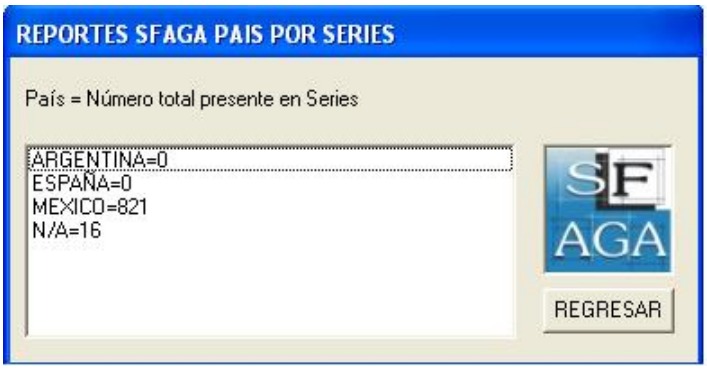


Figura 44. Ejemplo de los botones de reportes

También se generan otro tipo de reportes, los cuales pueden ser guardados en archivos de texto, que sirven como históricos, estos se centran principalmente en los préstamos realizados,

los números totales de series y programas y porque canales de transmisión se transmiten mas programas, un ejemplo de estos reportes se ve en la figura 45.

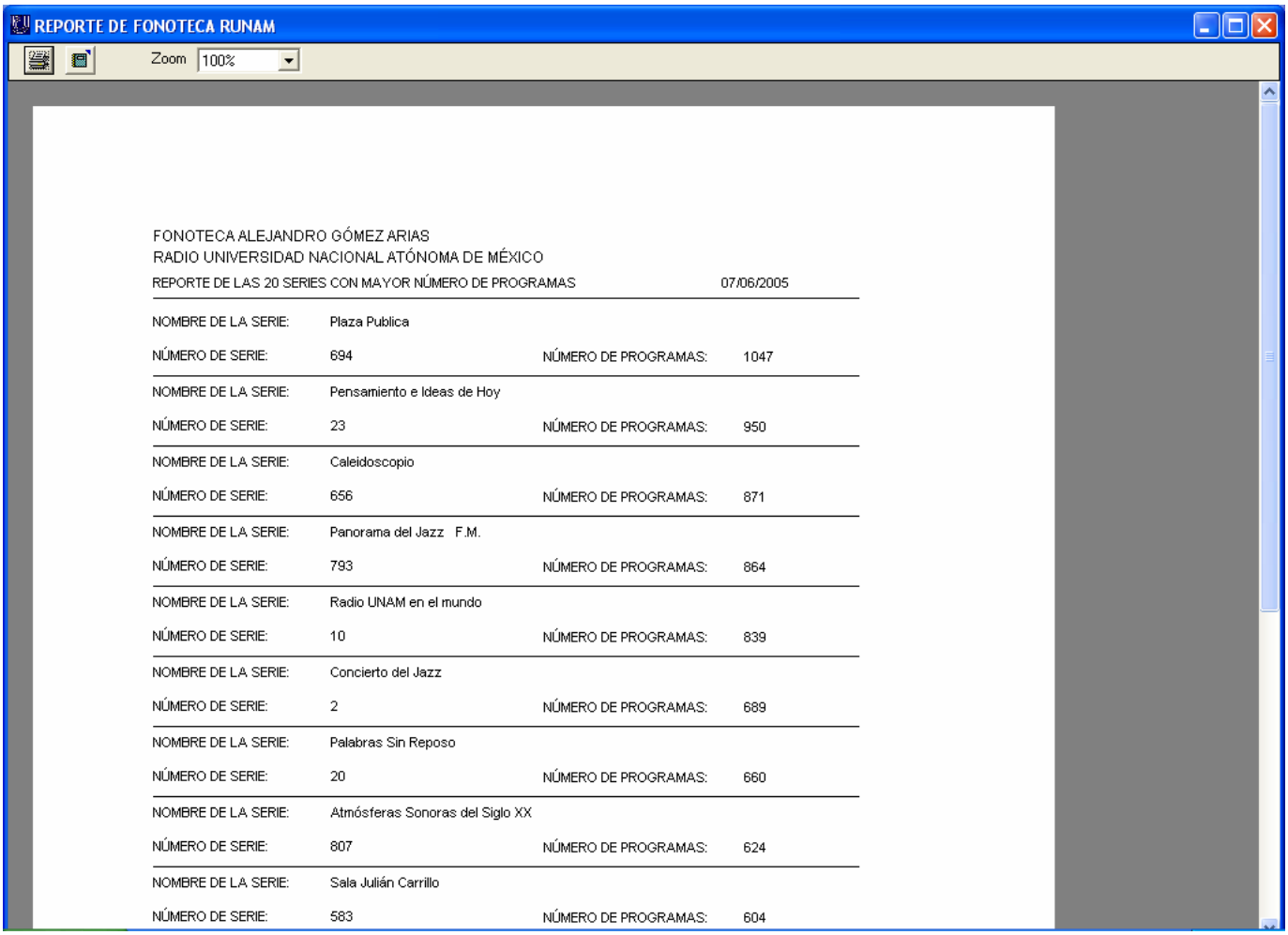


Figura 45. Ejemplo de reportes

Los dos últimos botones que se muestran son “Acerca” y “Salir”, que ya se mencionaron anteriormente.

Las tareas para eliminar registros de la base de datos, se realizan desde el analizador de consultas sql, en el cual se escriben instrucciones sql con las que se realizarán el borrado de los registros, estas instrucciones recibirán parámetros los cuales se tomaran de la back-end de la aplicación y serán proporcionados por el administrador (dba). A continuación se muestran los

archivos que contienen las instrucciones sql.

Archivo eliminar_serie.sql

```
/* ##     instrucciones sql para eliminar completamente una serie y sus respectivos programas
   ##   las tablas SERIES y PROGRAMAS tiene una relación fuerte, por lo que solo se podrá
   ##   eliminar una serie si esta no tiene asociado ningún programa.
   ##   de acuerdo a lo anterior, primero se elimina, en caso de existir, todos los programas de la serie,
   ##   y después se elimina la serie que corresponda con el id_series proporcionado.
   ##   El campo id_series es la llave primaria de la tabla SERIES y en la tabla PROGRAMAS es parte de la
   ##     llave primaria compuesta por: id_programas y id_series
         nota id_series es el número que corresponde a la serie
   ##  instrucciones  para eliminar completamente una serie y sus respectivos programas
   ##  ejemplo:
   ##  delete from PROGRAMAS where id_series=826
   ##  lo anterior eliminará todos los programas referentes a la serie número 826
   ##  delete from SERIES where id_series=826
   ##  lo anterior eliminará la serie correspondiente al número de serie 826
 !NOTA:una vez realizadas las instrucciones la información de la serie correspondiente con el número de  id_series proporcionado será eliminada de la base de datos, siendo la única opción de recuperar esta información  con un restore (recuperación) de un bakup (respaldo) previamente hecho desde la consola de administración de sql server, ver manual de usuario para la realizar estas tareas.
*/
delete from PROGRAMAS where id_series=número_serie /* ## sustituya número_serie por el valor correspondiente (valor entero) /*
delete from SERIES where id_series=número_serie     /* ## sustituya número_serie por el valor correspondiente (valor entero) /*
/* ##  este código debe ser ejecutado solo por el dba sql server, desde el analizador de consultas sql server:
   ##  inicio-> programas->Microsoft SQL Server-->Herramientas-->analizador de consultas sql-->Proporcionar Usuario y Password
   ##  de usuario con privilegios de administración sql-->Archivo-->Abrir--->
         Ruta_donde_se_encuntra_este_achivo.sql-->abrir
   ##  con este procedimiento se tendrá el contenido de este archivo en el analizador de consultas.
   ##  ahora se modifica el número_serie por el id_series correspondiente, se prueba la sintaxis del las instrucciones
   ##  sql Consulta-->analizar  si la salida en el analizador  es: Comandos completados con éxito. se procede a realizar
```

las instrucciones con Consulta--> ejecutar con lo que se elimina la serie de la base de datos. Y se muestran el número
de filas afectadas por las instrucciones.
si la salida en el analizador es similar: Servidor: mensaje 170, nivel 15, estado 1, línea 1 ,Línea 1: sintaxis incorrecta cerca de ''.
se debe consultar con el dba, para corregir errores de sintaxis.
finalmente cerrar analizador de consultas: Archivo-->salir-->guardar cambios a elimar_serie.sql-->no
*/

Archivo eliminar_programa.sql

```
/* ##     instrucciones sql para eliminar completamente un programa perteneciente a una serie
   ##   El campo id_series en la tabla PROGRAMAS es parte de la llave primaria compuesta por:
   ##   id_programas y id_series
   nota id_series es el número que corresponde a la serie y num_prog_serie es el número del programa en la  serie
   ##  instrucciones para eliminar completamente programas pertenecientes a una serie
   ##  ejemplo:
   ##  delete from PROGRAMAS where id_series=826 and num_prog_serie=23
   ##  lo anterior eliminará el programa número 23 de la serie 826
       !NOTA:una vez realizada la instrucción la información del programa de la serie correspondiente con el número
       de id_series y num_prog_serie proporcionados, será eliminada de la base de datos, siendo la única opción de recuperar esta información   es con un restore (recuperación) de un bakup (respaldo) previamente hecho, desde la consola de administración de sql  server, ver manual de usuario para la realizar estas tareas.
*/
delete from PROGRAMAS where (id_series=número_serie) and (num_prog_serie=número_programa)
/* ## sustituya número_serie y número_programa por los valores correspondientes (valores entero)
   ## si se requiere eliminar varios programas se puede seguir
  delete from PROGRAMAS where (id_series=número_serie) and (num_prog_serie=número_programa1) and (num_prog_serie=número_programa2) and ...
   ## donde número_programa1 es un programa de la serie, y número_programa2 es otro programa de la serie
   ## si se requiere eliminar programas con números consecutivos se puede seguir
  delete from PROGRAMAS where (id_series=número_serie) and (num_prog_serie > número_programa1) and (num_prog_serie < número_programa2)
  ## donde número_programa1 es el inicio del rango y número_programa2 es el fin del rango
  ##  símbolos aceptados:
  ##  símbolo      significado
  ##   >           mayor a
```

```
  ##   <          menor a
  ##   >=         mayor igual a
  ##   <=         menor igual a
  ##   <>         diferente a
*/
/* ##  este código debe ser ejecutado solo por el dba sql server, desde el analizador de consultas sql server:
   ##  inicio-> programas->Microsoft SQL Server-->Herramientas-->analizador de consultas sql-->Proporcionar
Usuario y Password
   ##  de usuario con privilegios de administración sql-->Archivo-->Abrir--->
Ruta_donde_se_encuntra_este_achivo.sql-->abrir
   ##  con este procedimiento se tendrá el contenido de este archivo en el analizador de consultas.
   ##  ahora se modifica el número_serie por el id_series correspondiente, se prueba la sintaxis del las
instrucciones
   ##  sql Consulta-->analizar  si la salida en el analizador  es: Comandos completados con éxito. se procede a
realizar
   ##  las instrucciones con Consulta--> ejecutar con lo que se eliminara el programa de la base de datos. Y se
muestran el número
   ##  de filas afectadas por las instrucciones.
   ##  si la salida en el analizador es similar:  Servidor: mensaje 170, nivel 15, estado 1, línea 1 ,Línea 1: sintaxis
incorrecta cerca de ''.
   ##  se debe consultar con el dba, para corregir errores de sintaxis.
   ##  finalmente cerrar analizador de consultas: Archivo-->salir-->guardar cambios a elimar_programa.sql-->no
*/
```

Archivo eliminar_programas_series.sql

```
/* ##     instrucciones sql para eliminar completamente programas pertenecientes a una serie
   ##   El campo id_series en la tabla PROGRAMAS es parte de la llave primaria compuesta por:
   ##   id_programas y id_series
        nota id_series es el número que corresponde a la serie y id_programas es el identificador númerico
       de los programas de la base de datos.
   ##  instrucciones para eliminar completamente programas pertenecientes a una serie
   ##  ejemplo:
   ##  delete from PROGRAMAS where id_series=826
   ##  lo anterior eliminará todos los programas referentes a la serie número 826

       !NOTA:una vez realizada la instrucción la información de los programas de la serie correspondiente con el
```

número de id_series proporcionado será eliminada de la base de datos, siendo la única opción de recuperar esta información es con un restore (recuperación) de un bakup (respaldo) previamente hecho, desde la consola de administración de sql server, ver manual de usuario para la realizar estas tareas.
*/
delete from PROGRAMAS where id_series=número_serie /* ## sustituya número_serie por el valor correspondiente (valor entero) /*
/* ## este código debe ser ejecutado solo por el dba sql server, desde el analizador de consultas sql server:
inicio-> programas->Microsoft SQL Server-->Herramientas-->analizador de consultas sql-->Proporcionar Usuario y Password
de usuario con privilegios de administración sql-->Archivo-->Abrir-->Ruta_donde_se_encuntra_este_achivo.sql-->abrir
con este procedimiento se tendrá el contenido de este archivo en el analizado de consultas.
ahora se modifica el número_serie por el id_series correpondiente, se prueba la sintaxis del las instrucciones
sql Consulta-->analizar si la salida en el analizador es: Comandos completados con éxito. se procede a realizar
las instrucciones con Consulta--> ejecutar con lo que se elmiana la serie de la base de datos. Y se muestran el número
de filas afectadas por las instrucciones.
si la salida en el analizador es similar: Servidor: mensaje 170, nivel 15, estado 1, línea 1 ,Línea 1: sintaxis incorrecta cerca de ''.
se debe consultar con el dba, para corregir errores de sintaxis.
finalmente cerrar analizador de consultas: Archivo-->salir-->guardar cambios a elimiar_programas_serie.sql-->no
*/

Los archivos anteriores estarán en la ruta donde se instalo el sistema SFAGA.

APÉNDICE C

En este apéndice se muestra parte del código sql generado por la herramienta Erwin correspondiente a la base de datos relacional para el desarrollo del sistema SFAGA de la Fonoteca de Radio UNAM.

```
/* código sql para la base de datos de la Foonoteca de Radio UNAM del proyecto SFAGA  */
/* Sistema de Automatización de la Fonoteca Alejandro Gómez Arias */
/* Creadores */
/* Ing. Hector Balcazar Ponce */
/* Ing. Oscar Ruelas Vela */
/* Fecha de ultima revisión 14 de agosto 2005 */

/* validacion de tablas existentes y objetos existentes

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[FK__INTERESAD__id_de__1273C1CD]') and
OBJECTPROPERTY(id, N'IsForeignKey') = 1)
ALTER TABLE [dbo].[INTERESADO] DROP CONSTRAINT FK__INTERESAD__id_de__1273C1CD
GO

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[FK_PROGRAMAS_PRESTAMO]') and OBJECTPROPERTY(id,
N'IsForeignKey') = 1)
ALTER TABLE [dbo].[PROGRAMAS] DROP CONSTRAINT FK_PROGRAMAS_PRESTAMO
GO

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[BUS_SER_PERIDIO]') and OBJECTPROPERTY(id, N'IsView') = 1)
drop view [dbo].[BUS_SER_PERIDIO]
GO

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[DEPINT]') and OBJECTPROPERTY(id, N'IsView') = 1)
drop view [dbo].[DEPINT]
GO

/* creacion de tablas

CREATE TABLE [dbo].[FORMATO] (
   [id_formato] [int] IDENTITY (1, 1) NOT NULL ,
   [tipo_formt_p] [varchar] (25) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[GRABACION] (
   [id_grabacion] [int] IDENTITY (1, 1) NOT NULL ,
   [tipo_grab_p] [varchar] (25) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[Hoja1$] (
   [id_series] [float] NULL ,
   [nombre_s] [nvarchar] (255) COLLATE Traditional_Spanish_CI_AS NULL ,
   [productor_s] [nvarchar] (255) COLLATE Traditional_Spanish_CI_AS NULL ,
   [numprog_s] [float] NULL ,
   [fecha_ini_s] [nvarchar] (255) COLLATE Traditional_Spanish_CI_AS NULL ,
   [resumen_s] [nvarchar] (255) COLLATE Traditional_Spanish_CI_AS NULL ,
   [vigencia_s] [nvarchar] (255) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ciudad_s] [nvarchar] (255) COLLATE Traditional_Spanish_CI_AS NULL ,
   [id_pais] [float] NULL ,
   [proyco_s] [nvarchar] (255) COLLATE Traditional_Spanish_CI_AS NULL ,
   [fecha_fin_s] [nvarchar] (255) COLLATE Traditional_Spanish_CI_AS NULL ,
   [id_idioma] [int] NULL ,
   [id_periodo] [int] NULL ,
   [id_publico] [int] NULL ,
   [id_caracter] [int] NULL ,
   [id_tp] [int] NULL ,
   [soportes_s] [int] NULL ,
   [capturo_s] [int] NULL ,
   [fecha_cap_s] [smalldatetime] NULL
```

ERWIN

```
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[IDIOMA] (
   [id_idioma] [int] IDENTITY (1, 1) NOT NULL ,
   [nombre_idioma] [varchar] (25) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[INTERESADO] (
   [id_interedado] [int] IDENTITY (1, 1) NOT NULL ,
   [nombre_i] [varchar] (120) COLLATE Traditional_Spanish_CI_AS NULL ,
   [id_depto] [int] NOT NULL ,
   [num_pres_i] [int] NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[PAIS] (
   [id_pais] [int] IDENTITY (1, 1) NOT NULL ,
   [nombre_pais] [varchar] (25) COLLATE Traditional_Spanish_CI_AS NULL ,
   [siglas_pais] [varchar] (5) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[PERIODO] (
   [id_periodo] [int] IDENTITY (1, 1) NOT NULL ,
   [tipo_per_s] [varchar] (25) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[PRESTAMO] (
   [id_prestamo] [int] IDENTITY (1, 1) NOT NULL ,
   [fecha_pres] [char] (10) COLLATE Traditional_Spanish_CI_AS NULL ,
   [fecha_dev] [varchar] (10) COLLATE Traditional_Spanish_CI_AS NULL ,
   [edo_pres] [int] NULL ,
   [id_interedado] [int] NULL ,
   [id_respons] [int] NULL ,
   [res_cierre] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
   [fecha_cierre] [char] (10) COLLATE Traditional_Spanish_CI_AS NULL ,
   [clave_pres] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
   [nombre_p_p] [varchar] (1024) COLLATE Traditional_Spanish_CI_AS NULL ,
   [edo_pres_t] [int] NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[PROGRAMAS] (
   [id_programa] [int] IDENTITY (1, 1) NOT NULL ,
   [id_series] [int] NOT NULL ,
   [nombre_p] [varchar] (512) COLLATE Traditional_Spanish_CI_AS NULL ,
   [conductor_p] [varchar] (120) COLLATE Traditional_Spanish_CI_AS NULL ,
   [fecha_codif_p] [varchar] (10) COLLATE Traditional_Spanish_CI_AS NULL ,
   [num_prog_serie] [int] NULL ,
   [fecha_cap_p] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
   [fecha_mod_p] [datetime] NULL ,
   [folio_p] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
   [fecha_grab_p] [datetime] NULL ,
   [tipo_prog] [varchar] (18) COLLATE Traditional_Spanish_CI_AS NULL ,
   [duracion_p] [varchar] (18) COLLATE Traditional_Spanish_CI_AS NULL ,
   [soportes_p] [int] NULL ,
   [res_grab_p] [varchar] (120) COLLATE Traditional_Spanish_CI_AS NULL ,
   [lugar_grab_p] [varchar] (60) COLLATE Traditional_Spanish_CI_AS NULL ,
   [resumen_p] [varchar] (1024) COLLATE Traditional_Spanish_CI_AS NULL ,
   [invitado_p] [varchar] (120) COLLATE Traditional_Spanish_CI_AS NULL ,
   [obs_p] [varchar] (254) COLLATE Traditional_Spanish_CI_AS NULL ,
   [modifico_p] [int] NULL ,
   [capturo_p] [int] NULL ,
   [edo_pres_p] [int] NULL ,
   [id_prestamo] [int] NULL ,
   [id_tp] [int] NULL ,
   [id_formato] [int] NULL ,
```

```
   [id_grabacion] [int] NULL ,
   [cont_pres] [int] NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[PUBLICO] (
   [id_publico] [int] IDENTITY (1, 1) NOT NULL ,
   [tipo_pub_s] [varchar] (25) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[RESPONSABLE] (
   [id_respons] [int] NOT NULL ,
   [nombre_r] [varchar] (120) COLLATE Traditional_Spanish_CI_AS NULL ,
   [puesto_r] [varchar] (60) COLLATE Traditional_Spanish_CI_AS NULL ,
   [contrasena_r] [char] (8) COLLATE Traditional_Spanish_CI_AS NULL ,
   [direccion_r] [varchar] (120) COLLATE Traditional_Spanish_CI_AS NULL ,
   [telefono_r] [varchar] (20) COLLATE Traditional_Spanish_CI_AS NULL ,
   [fecha_alta_r] [varchar] (10) COLLATE Traditional_Spanish_CI_AS NULL ,
   [clave_r] [varchar] (10) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[SERIES] (
   [id_series] [int] NOT NULL ,
   [clasi_s] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
   [nombre_s] [varchar] (512) COLLATE Traditional_Spanish_CI_AS NULL ,
   [folio_s] [char] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
   [numprog_s] [int] NULL ,
   [tema_s] [varchar] (254) COLLATE Traditional_Spanish_CI_AS NULL ,
   [productor_s] [varchar] (120) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ciudad_s] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
   [proyco_s] [varchar] (120) COLLATE Traditional_Spanish_CI_AS NULL ,
   [fecha_ini_s] [datetime] NULL ,
   [fecha_fin_s] [datetime] NULL ,
   [fecha_cap_s] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
   [fecha_mod_s] [datetime] NULL ,
   [fecha_codif_s] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
   [tiempo_s] [varchar] (18) COLLATE Traditional_Spanish_CI_AS NULL ,
   [localizacion_s] [varchar] (60) COLLATE Traditional_Spanish_CI_AS NULL ,
   [rubrica_s] [varchar] (254) COLLATE Traditional_Spanish_CI_AS NULL ,
   [resumen_s] [varchar] (1024) COLLATE Traditional_Spanish_CI_AS NULL ,
   [modifico_s] [int] NULL ,
   [capturo_s] [int] NULL ,
   [vigencia_s] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
   [id_pais] [int] NULL ,
   [id_publico] [int] NULL ,
   [id_periodo] [int] NULL ,
   [id_idioma] [int] NULL ,
   [num_soprt_s] [int] NULL ,
   [id_tp] [int] NULL ,
   [id_caracter] [int] NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[TRANSMSION] (
   [id_tp] [int] IDENTITY (1, 1) NOT NULL ,
   [ft] [datetime] NULL ,
   [ht] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ct] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ft1] [datetime] NULL ,
   [ht1] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ct1] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ft2] [datetime] NULL ,
   [ht2] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ct2] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ft3] [datetime] NULL ,
   [ht3] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ct3] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ft4] [datetime] NULL ,
```

ERWIN

```
   [ht4] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ct4] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ft5] [datetime] NULL ,
   [ht5] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ct5] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ft6] [datetime] NULL ,
   [ht6] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ct6] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ft7] [datetime] NULL ,
   [ht7] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
   [ct7] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
   [veces_trans] [int] NULL
) ON [PRIMARY]
GO

SET QUOTED_IDENTIFIER ON
GO
SET ANSI_NULLS ON
GO

/* fin de creacion de tablas

/* creacion de vistas

CREATE VIEW dbo.BUS_SER_PERIDIO
AS
SELECT     dbo.SERIES.id_series, dbo.SERIES.nombre_s, dbo.AUX.id_tibus, dbo.AUX.pala_busqueda, dbo.PERIODO.tipo_per_s,
dbo.CARACTER.tipo_car_s,
                      dbo.IDIOMA.nombre_idioma
FROM         dbo.SERIES INNER JOIN
                      dbo.PERIODO ON dbo.SERIES.id_periodo = dbo.PERIODO.id_periodo INNER JOIN
                      dbo.CARACTER ON dbo.SERIES.id_caracter = dbo.CARACTER.id_caracter INNER JOIN
                      dbo.IDIOMA ON dbo.SERIES.id_idioma = dbo.IDIOMA.id_idioma CROSS JOIN
                      dbo.AUX

GO
SET QUOTED_IDENTIFIER OFF
GO
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER OFF
GO
SET ANSI_NULLS ON
GO

/* se crean las VISTAS   */

CREATE VIEW dbo.DEPINT
AS
SELECT dbo.INTERESADO.id_interedado,
   dbo.INTERESADO.nombre_i, dbo.DEPTO.nombre_d
FROM dbo.INTERESADO INNER JOIN
   dbo.DEPTO ON
   dbo.INTERESADO.id_depto = dbo.DEPTO.id_depto

GO
SET QUOTED_IDENTIFIER OFF
GO
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER OFF
GO
SET ANSI_NULLS ON
GO

CREATE VIEW dbo.[FROM VER_PROGPRES_DEUDA]
AS
SELECT PROGRAMAS.nombre_p, PROGRAMAS.edo_pres_p,
```

```
   PRESTAMO.fecha_dev, PRESTAMO.edo_pres
FROM PROGRAMAS, PRESTAMO
WHERE PRESTAMO.id_prestamo = PROGRAMAS.id_prestamo AND
   (PROGRAMAS.edo_pres_p = 1) AND
   (PRESTAMO.edo_pres = 1)

GO
SET QUOTED_IDENTIFIER OFF
GO
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER OFF
GO
SET ANSI_NULLS ON
GO


CREATE VIEW dbo.IMPRESION
AS
SELECT dbo.SERIES.nombre_s, dbo.PROGRAMAS.nombre_p,
   dbo.PROGRAMAS.folio_p, dbo.PROGRAMAS.duracion_p,
   dbo.PROGRAMAS.obs_p, dbo.PROGRAMAS.res_grab_p,
   dbo.PROGRAMAS.num_prog_serie,
   dbo.SERIES.id_series
FROM dbo.SERIES INNER JOIN
   dbo.PROGRAMAS ON
   dbo.SERIES.id_series = dbo.PROGRAMAS.id_series

GO
SET QUOTED_IDENTIFIER OFF
GO
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO
SET ANSI_NULLS ON
GO

CREATE VIEW dbo.MAX_PREST
AS
SELECT     MAX(id_prestamo) AS MAX_PRES
```

BIBLIOGRAFÍA

BIBLIOGRAFÍA

Bibliografía

Roger S. Presuman
Ingeniería del Software, Un enfoque práctico
Ed. McGraw-Hill, 1998

Korth H. Sliberschatz A.
Fundamentos de Bases de Datos.
Ed. McGraw-Hill, 1993

C. Batín, S. Cerí, S. B. Navathe (1994)
Diseño Conceptual de Bases de Datos, Un enfoque de entidades-interrelaciones.
Addison-Wesley / Díaz de Santos

T. Connolly, C. Begg, A. Strachan (1996)
Database Systems. A Practical Approach to Design, Implementation and Management
Addison-Wesley
Segunda Edición en 1998

C. J. Date (1993)
Introducción a los sistemas de Bases de Datos
Volumen I, Quinta Edición
Addison-Wesley Iberoamericana
Sexta edición en 1995 (en inglés, por Addison-Wesley)

R. Elmasri, S. B. Navathe (1997)
Sistemas de Bases de Datos, Conceptos fundamentales
Segunda Edición
Addison-Wesley Iberoamericana
Tercera edición en 1999 (en inglés, por Addison-Wesley)

M. J. Fol., B. Zoellick (1992)
File Structures
Segunda Edición
Addison-Wesley

G. W. Hansen, J. V. Hansen (1997)
Diseño y Administración de Bases de Datos
Segunda Edición
Prentice may

Aguilar R. Francisco, García A. Claudia, Monzón P. Rafael, Pérez G. Eduardo (2003)
Tesis Ingeniería en Computación “Sistema de Control de Servicios y Emergencias para una Empresa Purificadora de Agua”
Facultad de Ingeniería.

M. J. Hernández (1997)
Database Desing for Mere Mortals
Addison-Wesley Developers Press

Tanenbaum. Andrew. S.
Redes de Ordenadores
Prentice-Hall Hispanoamericana
Segunda Edición, 1991.

Comer E., Douglas.
Redes globales de información con Internet y TCP/IP
Tercera Edición.
Prentice Hall, 1996

Stallings, William
Comunicaciones y redes de computadores
Quinta Edición.
Prentice Hall, 1997