UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

"REINGENIERÍA DEL SISTEMA INTEGRAL DE TESORERÍA GENERAL DE LA ALDF"

T E S I S

QUE PARA OBTENER EL TÍTULO DE
INGENIERO EN COMPUTACIÓN
P R E S E N T A :

ARMANDO MONTIEL PATIÑO

DIRECTOR DE TESIS:

ING. ALBERTO TEMPLOS CARBAJAL

CIUDAD UNIVERSITARIA. **ABRIL DE 2015**

AGRADECIMIENTOS

A la Universidad Nacional Autónoma de México, mi alma mater y en especial a la Facultad de Ingeniería, por haberme dado la oportunidad de cursar una de las carreras más actuales, apasionantes y satisfactorias que pude haber elegido.

A ti mamá, gracias por todo tu esfuerzo, tu apoyo y la confianza que depositas en mí. Gracias por haberme mostrado el camino para ser una persona de bien, tanto en lo personal como en lo profesional. Gracias por todos los momentos que hemos vivido y convivido juntos. Y gracias por haberme dado la vida. Te amo mucho.

A ti Adriana Barbosa Navarro, mi querida esposa, por tu amor, por tu apoyo, por tu ejemplo, por tu confianza en mí y por motivarme e impulsarme a concluir la presente. Te amo mucho.

A ti hijita Samar Montiel Barbosa, por echarme porras para continuar avanzando y creciendo tanto en lo personal, como en lo profesional. Te amo.

A ti hermanita Isabel por tu amor, tu apoyo y creer en mí en todo momento. Te amo.

A ti papá por ser un ejemplo de tenacidad y constancia. Te amo.

A todos mis amigos en general por todos los momentos que hemos pasado juntos tanto en las buenas como en las malas.

A todos mis maestros de toda la vida, tanto aquellos que me enseñaron en las aulas como fuera de ellas, porque forman parte de lo que soy ahora.

Al Ing. Alberto Templos Carvajal, mi director de tesis, por la confianza y la paciencia que tuviste conmigo para la conclusión de la presente.

A mis sinodales por el tiempo dedicado a mi persona y por darme la oportunidad de demostrarles que lo que aprendí en nuestra Facultad de Ingeniería, he sabido aplicarlo en el ámbito de la Ingeniería en Computación.

Armando Montiel Patiño.

ÍNDICE

1 Introducción 1
1.1 Planteamiento del problema 1
1.2 Objetivo 2

2 Ingeniería de software 3
2.1 Metodología 4
2.2 Reingeniería de software 4
2.3 Ingeniería de requisitos 5
2.4 Análisis de inventarios 7
2.5 Reestructuración de documentos 7
2.6 Ingeniería inversa 7
2.7 Reestructuración de código 8
2.8 Reestructuración de datos 9
2.9 Ingeniería del diseño 10
2.10 Prácticas de ingeniería de software 13

3 Definición del negocio 15
4 Ingeniería de requisitos 16
5 Identificación de procesos 17
6 Análisis de inventarios 18
7 Evaluación de procesos 30
8 Ingeniería inversa y reestructuración de documentos 31
9 Reestructuración de código y datos 37
10 Ingeniería del diseño 39
11 Especificación y diseño de procesos 53
12 Elaboración de prototipos 63
13 Refinamiento e instanciación (personalización) 63
14 Desarrollo Rápido de Aplicaciones (DRA) 71

CONCLUSIONES 81

BIBLIOGRAFÍA 83

1 INTRODUCCIÓN

En la actualidad el manejo de la información es un factor determinante en el desarrollo de toda organización. La necesidad de contar con herramientas informáticas que permitan al personal simplificar el trabajo administrativo y al mismo tiempo atender otras actividades relevantes se deriva de la obligación de una empresa de proporcionar atención rápida, confiable y segura, brindando un buen servicio a sus clientes.

Los grandes volúmenes de información y la tecnología que evoluciona vertiginosamente, han originado la creación de sistemas de cómputo y comunicaciones que facilitan la integración de actividades dentro de las empresas, así como el intercambio de datos en lugares distantes. Es imposible pensar que pueda existir una organización sin la utilización de la computadora como herramienta de apoyo en el procesamiento de información.

Aunque existen grandes firmas desarrolladoras de software, es importante hacer notar que éstas producen sistemas que ayudan a una empresa sólo en algunas tareas, sin embargo, debido a la complejidad de sus actividades, es necesario automatizar completamente el manejo de su información y englobar cada detalle dentro de un sistema de cómputo, siendo necesario desarrollar software a la medida, dejando el software comercial ya existente en el mercado para actividades rutinarias no específicas como son los procesadores de texto, hojas de cálculo, graficadores, etcétera. Sin embargo, el software hecho a la medida puede requerir ajustes, mejorías y procesos adicionales, derivado de las nuevas políticas y normatividades que la empresa requiera, sin la necesidad de desarrollar un nuevo sistema, tomando como base el existente. Por ello es necesario considerar la Reingeniería de Sistemas como una opción viable para el desarrollo de una nueva versión de un sistema existente.

1.1 PLANTEAMIENTO DEL PROBLEMA

Se cuenta con un Sistema hecho a la medida en 1997, desarrollado en Informix 4gl, con un motor de base de datos Informix, de acuerdo a las necesidades existentes en aquél momento, el cual sólo contaba con los módulos de Nómina, Pagos y Presupuesto, a los que en 2002 se le anexó el módulo de Contabilidad estando desarrollado en Delphi y Microsoft Access con conexión a un motor de base de datos Informix de tal manera que fue fácil su integración, el cual fue donado por la cámara de Senadores. Sin embargo, hubo cambios en los requerimientos de los diferentes módulos del sistema.

Módulo de Nómina: En el área de nóminas cambió la forma de calcular el impuesto de acuerdo a la Ley de Impuestos Sobre la Renta; los cálculos de Seguro de Retiro, Retiro/Cesantía/Edad avanzada/Vejez, Cuota ISSSTE y Préstamos hipotecarios que se efectúan al ISSSTE y al FOVISSSTE a través de la CONSAR; no se cuenta con un Kardex de empleados para registrarlos con un solo número y saber si cuando ingresan a la Asamblea se trata de un reingreso o una alta nueva y tener todo su historial de puestos, áreas de adscripción y fechas en las que estuvo activo; no se cuenta con la captura y cálculo del tiempo extra de los empleados de base en el que se desglose la parte gravada y la parte exenta, así como guardarlo en forma de historial detallado para aclaraciones; no se cuenta con el registro de faltas indicando número de días, no se cuenta con un registro de préstamos en general para llevar un contador de cuotas o letras pagadas quincena a quincena; no se cuenta con el Pago a Terceros que salen directamente de los conceptos de descuento de la nómina como son Pensión Alimenticia, Seguro Colectivo de Retiro, Seguro de Separación Individualizada, Aportaciones a Partidos Políticos, Cuotas Sindicales de tres Sindicatos, Seguro de Vida

Individual, Seguro de Gastos Médicos, Pago a las Bancas por financiamiento automotriz, Pago a Aseguradoras por Seguro automotriz y Pagos de Servicios Testamentarios, pagos que se efectuarán a través de cheques y que deben tener una interfaz con el módulo de Pagos, así como la generación de las Cuentas por Pagar correspondientes en el módulo de Presupuestos; no se cuenta con la generación automática de las Cuentas por Pagar al módulo de Presupuestos de las Nóminas procesadas, evitando su captura; y tampoco se cuenta con una bitácora que permita monitorear las actividades de los usuarios dentro del sistema para tener una mejor administración de los procesos que se operan.

Módulo de Pagos: Se requiere que la impresión de cheques incluya en la parte de la póliza del mismo, la impresión de las cuentas contables correspondientes a las que afectará en la contabilidad. En la protección de cheques se requiere que se genere un archivo plano para enviar al banco indicando los números y montos de éstos. Las dispersiones bancarias correspondientes a las nóminas se requiere que se generen en archivos planos y que solo incluya al personal que tiene cuenta bancaria para tal efecto. En el Pago a Terceros que se genere automáticamente la Cuenta por Pagar correspondiente en el módulo de Presupuestos y de los cheques emitidos se generen automáticamente las Pólizas de egresos en el módulo de Contabilidad, evitando su captura.

Módulo de Presupuestos: Se requiere que se generen automáticamente las Cuentas por Pagar correspondientes, de acuerdo a su origen en los módulos de Nómina y Pagos para evitar su captura manual y consecuentes errores y pérdida de tiempo.

Módulo de Contabilidad: Se requiere que se generen automáticamente las Pólizas correspondientes, de acuerdo a su origen en los módulos de Nómina, Presupuestos y Pagos para evitar su captura manual. Así mismo, se requiere incluir en cada Póliza el número de la Cuenta por Pagar que le dio origen, para poder ser analizadas más fácilmente en el reporte de Auxiliares.

1.2 OBJETIVO

Contar con una herramienta expedita que permita a los usuarios obtener los productos de éste en un mucho menor tiempo, realizando sus labores de una manera más sencilla y más eficiente, dejando al sistema a que realice otros procesos de manera automática, libre de errores humanos y pérdida de tiempo al capturar manualmente cuentas por pagar, cheques, pólizas contables y otra información dentro de los diferentes módulos del sistema, al igual que la generación de los archivos de envío a terceras instituciones para el intercambio de información como ISSSTE, FOVISSSTE, SAR y la Banca, y facilitar al usuario en la captura de otros datos haciendo más sencilla su interfaz y dejando que el sistema haga los procesos pesados y los cálculos correspondientes. Asimismo que el usuario final cuente con una herramienta que le permita analizar la información de manera expedita, pudiendo entregar a sus instancias superiores los reportes que se integran a la página de Internet, y otros reportes no planeados como los que se solicitan a través de Información Pública.

2 INGENIERÍA DE SOFTWARE

La ingeniería de software es el campo de la ciencia de la computación que trata con la construcción de sistemas de software que son tan grandes y complejos que son construidos por un equipo o equipos de ingenieros. Usualmente, estos sistemas de software existen en múltiples versiones y son usados por muchos años. Durante su ciclo de vida sufren muchos cambios; para corregir defectos, mejorar características existentes, añadir nuevas características, quitar viejas características, o ser adaptados a correr en un nuevo ambiente.

Parnas definió la ingeniería de software como "la construcción de software multi-versiones por multi-personas". Esta definición captura la esencia de la ingeniería de software y remarca las diferencias entre programación y la ingeniería de software. Un programador escribe un programa completo, mientras que un ingeniero de programación escribe un componente de software que será combinado con componentes escritos por otros ingenieros de programación para construir un sistema. El componente que uno escribe puede ser modificado por otros; puede ser usado por otros para construir diferentes versiones del sistema tiempo después de que el primero haya dejado el proyecto. La programación es primeramente una actividad personal, mientras que la ingeniería de software es esencialmente una actividad de equipo.

El término "ingeniería de software" fue inventado a finales de los años 1960s después de darse cuenta de que todas las lecciones aprendidas acerca de cómo programar bien no ayudaría a construir mejores sistemas de software. Mientras que el campo de la programación ha hecho tremendos progresos (a través del estudio sistemático de algoritmos, estructuras de datos y la invención de la "programación estructurada"), ha tenido todavía grandes dificultades en construir sistemas de software grandes. Las técnicas que fueron usadas por un físico escribiendo un programa para calcular la solución de una ecuación diferencial para un experimento no fueron adecuadas para un programador trabajando en un equipo que estaba tratando de construir un sistema operativo o quizá un sistema de seguimiento de inventario. Lo que se necesitaba en este caso complejo era el enfoque clásico de la ingeniería: definir claramente el problema que se estaba tratando de resolver, y desarrollar herramientas y técnicas estándar para resolverlo.

La ingeniería de software ha ido progresando desde los años 1960s. Existen técnicas estándar que son usadas en el campo. Pero el campo está aún lejos de ser catalogada en el status de una clásica disciplina de ingeniería. Muchas áreas permanecen en el campo que todavía están siendo enseñadas y practicadas con base en técnicas informales. Generalmente aún no hay métodos aceptados para especificar que es lo que un sistema de software debe hacer. En el diseño de un sistema eléctrico, como puede ser un amplificador, el sistema está especificado precisamente. Todos los parámetros y niveles de tolerancia están definidos claramente y son comprendidos por el cliente y el ingeniero. En la ingeniería de software, estamos empezando a definir qué tantos parámetros deben de haber para un sistema de software y (una tarea aparentemente mucho más difícil) cómo especificarlos.

Además, en las disciplinas de la ingeniería clásica, la ingeniería está equipada con herramientas y la madurez matemática para especificar las propiedades del producto separadamente de aquellas del diseño. Por ejemplo, un ingeniero eléctrico confía en las ecuaciones matemáticas para verificar que el diseño no violará los requerimientos de potencia. En la ingeniería de software, esas herramientas matemáticas no han sido bien desarrolladas. El típico ingeniero de programación confía mucho más en la experiencia y el juicio, más que en técnicas matemáticas. Mientras que la experiencia y el juicio son necesarios, las herramientas de análisis formal también son esenciales en la práctica de la ingeniería.

En este trabajo se verá que la ingeniería de software debe ser practicada como una disciplina de ingeniería. Mi enfoque es presentar ciertos principios en los que creo que son esenciales y más importantes que cualquier notación o metodología, en particular para la construcción de sistemas de software. Existen diferentes metodologías y cada una se puede aplicar cuando sean apropiadas de acuerdo al proyecto. Para este caso en particular he elegido la Reingeniería de Software.

2.1 METODOLOGÍA

Debido a que la mayor parte del Sistema Integral de la Tesorería es funcional, se utilizará la metodología de Reingeniería de Software en la cual se aplicará la Ingeniería de Requisitos para entender las necesidades de los usuarios, el Análisis de Inventarios, la Reestructuración de Documentos, la Ingeniería Inversa, la Reestructuración de Código y la Reestructuración de Datos para entender y optimizar el funcionamiento de los procesos existentes, la Ingeniería del Diseño, y las prácticas de la Ingeniería de Software utilizando el modelo de Desarrollo Rápido de Aplicaciones para la mayoría de los nuevos procesos, ya que varios de ellos son semejantes y sus productos se obtienen de los mismos procesos que dan lugar a la información requerida.

2.2 REINGENIERÍA DE SOFTWARE

La reingeniería es una actividad de reconstrucción y readaptación de un sistema a nuevas tareas sin tener que desarrollar un nuevo sistema, inspeccionando éste para determinar si es factible la reconstrucción del mismo. Para implementarlo se aplica un modelo de reingeniería del software que se definen las **seis** actividades mostradas en la figura 2.1. En ocasiones, estas actividades se producen de forma secuencial y lineal, pero esto no siempre es así. Por ejemplo, puede ser que la ingeniería inversa tenga que producirse antes de que pueda comenzar la reestructuración de documentos.

El paradigma de la reingeniería mostrado en la figura 2.1 (modelo de **Reingeniería de Procesos de Negocios**) es un modelo cíclico. Esto significa que cada una de las actividades mostradas como parte del paradigma pueden repetirse en otras ocasiones. Para un ciclo en particular, el proceso puede terminar después de cualquiera de estas actividades.

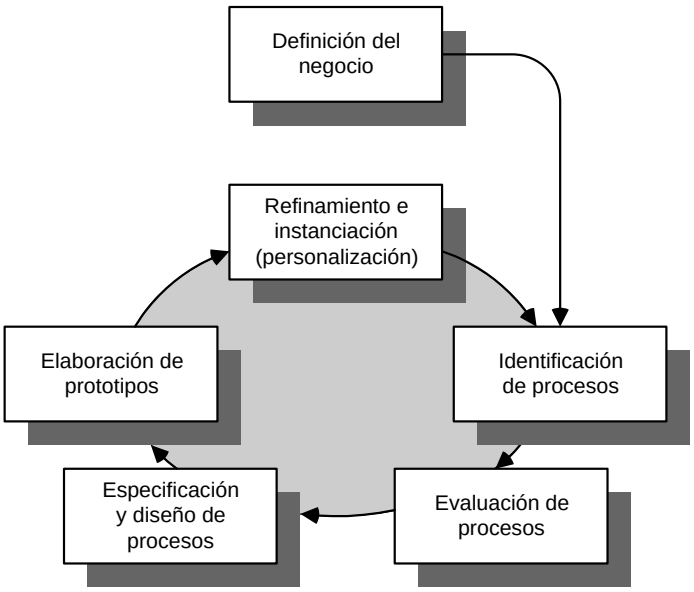


FIGURA 2.1 Modelo de RPN

Definición del negocio. Las metas del negocio se identifican dándole el contexto de cuatro controladores clave: reducción de costo, reducción de tiempos, mejora de calidad y desarrollo, y fortalecimiento de personal. Es posible definir las metas al nivel del negocio o respecto de un componente específico del negocio.

Identificación de procesos. Se identifican los procesos cruciales para lograr las metas precisadas en la definición del negocio, luego podría clasificarse de acuerdo con su importancia, necesidad de cambio o en cualquier otra forma que sea adecuada para la actividad de reingeniería.

Evaluación de procesos. El proceso existente se analiza y mide exhaustivamente. Se identifican las tareas del proceso; se anotan los costos y el tiempo que consumen las tareas del proceso; y se aíslan los problemas de calidad y desempeño.

Especificación y diseño de procesos. Con base en la retroalimentación obtenida durante las primeras tres actividades de la RPN, se preparan casos de uso para cada proceso que será rediseñado. En el contexto de la RPN los casos de uso identifican un escenario que entrega cierto resultado a un cliente. Con el caso de uso como la especificación del proceso se diseña un nuevo conjunto de tareas para el proceso.

Elaboración de prototipos. Un proceso de negocio rediseñado debe convertirse en prototipo antes de que sea integrado por completo en el negocio. Esta actividad “prueba” el proceso de modo que puedan llevarse a cabo refinamientos.

Refinamiento e instanciación (personalización). Con base en la retroalimentación del prototipo, el proceso de negocio se refina y luego se particulariza dentro de un sistema de negocio.

2.3 INGENIERÍA DE REQUISITOS

La ingeniería de requisitos, como todas las demás actividades de la ingeniería del software, debe adaptarse a las necesidades del proceso, el proyecto, el producto y las personas que realizan el trabajo. Desde la perspectiva del proceso del software, la ingeniería de requisitos es una acción de la ingeniería de software que comienza durante la actividad de comunicación y continúa en la actividad de modelado. En esta etapa es esencial que el equipo de software haga un esfuerzo real por entender los requisitos de un problema antes de intentar resolverlo.

La ingeniería de requisitos proporciona el mecanismo apropiado para entender lo que el cliente quiere, analizar las necesidades, evaluar la factibilidad, negociar una solución razonable, especificar la solución sin ambigüedades, validar la especificación, y administrar los requisitos, conforme éstos se transforman en un sistema operacional. El proceso de la ingeniería de requisitos se lleva a cabo a través de siete diferentes funciones: inicio, obtención, elaboración, negociación, especificación, validación y gestión.

Algunas de estas funciones de la ingeniería de requisitos ocurren en paralelo; todas deben adaptarse a las necesidades del proyecto, están dirigidas a definir lo que el cliente quiere, y sirven para establecer una base sólida respecto del diseño y la construcción de lo que obtendrá el cliente.

Inicio. Los proyectos generalmente comienzan con una serie de preguntas libres de contexto. El objetivo es establecer una comprensión básica del problema, las personas que

quieren una solución, la naturaleza de la solución que se desea, y la efectividad de la comunicación preliminar entre el cliente y el desarrollador.

Obtención. Un equipo de participantes y desarrolladores trabajan juntos para identificar el problema, proponer elementos de solución, negociar diferentes enfoques y especificar un conjunto preliminar de requisitos para la solución, aplicando alguna variación de las siguientes directrices básicas:

- Las reuniones las dirige alguno de los asistentes, ya sea un ingeniero de software o un cliente (junto con otros participantes interesados).
- Se establecen reglas para la preparación y la participación.
- Se sugiere una agenda que sea tan formal como para cubrir todos los puntos importantes, pero tan informal como para estimular el flujo de ideas.
- Un moderador (puede ser un cliente, un desarrollador o un agente externo) es quién controla la reunión.
- Se utiliza un "mecanismo de definición" (pueden ser hojas de trabajo, gráficos, hojas adheribles, un tablero electrónico, un mensajero electrónico o un foro virtual).
- La meta es identificar el problema, proponer elementos de solución, negociar diferentes enfoques y especificar un conjunto de requisitos de solución preliminares en una atmósfera que conduzca al cumplimiento de la meta.

Elaboración. La información conseguida con el cliente durante el inicio y la obtención se expande y se refina durante la elaboración. Esta actividad de la ingeniería de requisitos se enfoca en el desarrollo de un modelo técnico refinado de las funciones, características y restricciones del software.

La elaboración es una acción del modelado del análisis y se compone de una serie de tareas de modelado y refinamiento. La elaboración se conduce mediante la creación y el refinamiento de escenarios del usuario que describen la forma en que el usuario final interactúa con el sistema. Cada escenario del usuario se analiza para obtener clases de análisis: entidades del dominio de negocios visibles para el usuario final. Se definen los atributos de cada clase de análisis y se identifican los servicios que requieren cada clase. Se identifican las relaciones y la colaboración entre las clases y se produce una variedad de diagramas complementarios.

El resultado final de la elaboración es un modelo de análisis que define el dominio de la información, las funciones y el comportamiento del problema.

Negociación. Es usual que a veces los usuarios propongan requisitos que entran en conflicto entre sí, por lo que el ingeniero de requisitos debe conciliar estos conflictos por medio de un proceso de negociación. Se les pide a los usuarios y otros interesados que ordenen sus requisitos y después discutan los conflictos relacionados con la prioridad. Se identifican y analizan los riesgos asociados con cada requisito. Se hacen estimaciones preliminares del esfuerzo para su desarrollo y después se utilizan para evaluar el impacto de cada requisito en el costo del proyecto y sobre el tiempo de entrega. Mediante un enfoque iterativo, los requisitos se eliminan, combinan o modifican de forma que cada parte alcance cierto grado de satisfacción.

Especificación. A veces se requiere desarrollar y utilizar una plantilla estándar para una especificación, ya que esto lleva a que los requisitos sean presentados de una manera más consistente y por ende más entendible. Sin embargo, algunas veces es necesario ser flexible mientras se desarrolla una especificación. En relación a sistemas grandes el mejor enfoque podría ser un documento escrito que combinara descripciones en lenguaje natural y modelos

gráficos. Por otro lado, en cuanto a productos o sistemas más pequeños, podría ser que no se necesite más que escenarios de uso, cuando dichos sistemas residan en ambientes técnicos que se comprendan bien.

La especificación es el producto de trabajo final que genera la ingeniería de requisitos. Sirve como base para las actividades de ingeniería de software subsecuentes. Describe la función y el desempeño de un sistema basado en computadoras y las restricciones que generan su desarrollo.

Validación. La calidad de los productos de trabajo procedentes de la ingeniería de requisitos se evalúa durante un paso de validación. La validación de requisitos examina la especificación para asegurar que todos los requisitos de software se han establecido de manera precisa, que se han detectado las inconsistencias, omisiones y errores y que éstos han sido corregidos, y que los productos de trabajo cumplen con los estándares establecidos para el proceso, proyecto y producto.

Gestión de requisitos. Los requisitos para los sistemas basados en computadoras cambian y el deseo por cambiarlos persiste durante la vida del sistema. La gestión de requisitos es un conjunto de actividades que ayudan al equipo de proyecto a identificar, controlar y rastrear los requisitos y los cambios a éstos en cualquier momento mientras se desarrolla el proyecto.

La gestión formal de requisitos se inicia sólo para proyectos grandes, los cuales tienen cientos de requisitos identificables. En proyectos pequeños esta función de la ingeniería de requisitos es bastante menos formal.

2.4 ANÁLISIS DE INVENTARIOS

Se deberá disponer de un inventario de todas las aplicaciones que conforman el sistema. El inventario puede que no sea más que una lista con la información que proporciona una descripción detallada de todas las aplicaciones activas.

Las candidatas a la reingeniería serán aquellas que tengan una mayor importancia para la organización y deberán ser revisadas periódicamente debido a que el estado de las aplicaciones puede cambiar en función del tiempo, y como resultado, cambiarán también las prioridades para la reingeniería.

2.5 REESTRUCTURACIÓN DE DOCUMENTOS

Una documentación escasa es un hecho en muchos sistemas heredados, sin embargo, quizá no sea necesario documentarlo por completo, más bien se documentarán aquellas aplicaciones del sistema que estén siendo modificadas, incluso un enfoque inteligente es reducir la documentación al mínimo necesario.

2.6 INGENIERÍA INVERSA

La ingeniería inversa del software es el proceso del análisis de un programa, con el fin de crear una representación del programa con un nivel de abstracción más elevado que el código fuente. La ingeniería inversa es un proceso de recuperación del diseño, del proceso y de los datos de un programa, para descubrir los “secretos” que en él se encierran.

En la figura 2.2 se representa el proceso de ingeniería inversa. Antes de que comiencen las actividades de ingeniería inversa, el código fuente no estructurado (“sucio”) se reestructura de modo que solo contenga las estructuras de programación estructurada. Esto facilita la lectura del código fuente y ofrece la base para las subsecuentes actividades de ingeniería inversa.

El núcleo de la ingeniería inversa es una actividad llamada extracción de abstracciones. Se debe evaluar el programa antiguo, y a partir del código fuente (con frecuencia sin documentar), desarrollar una especificación significativa del procesamiento que realiza, la interfaz del usuario que se aplica y las estructuras de datos del programa o las bases de datos que se utilizan.

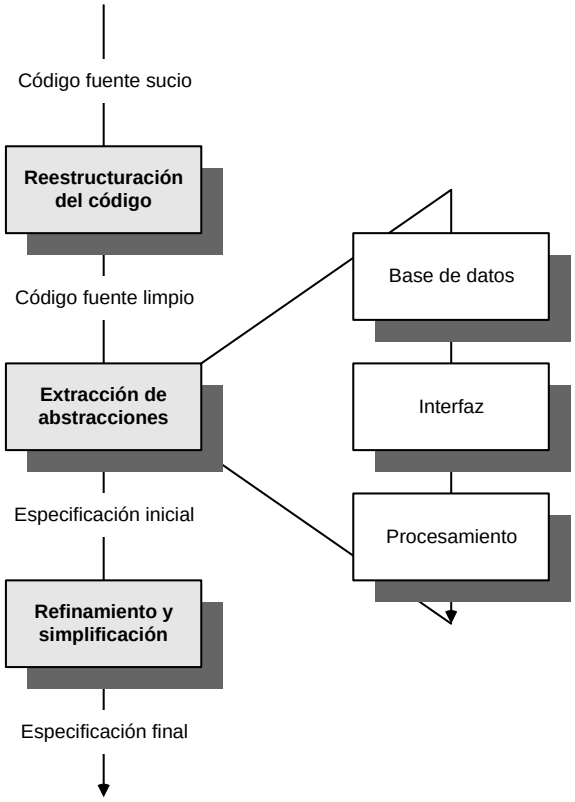


FIGURA 2.2 Proceso de ingeniería inversa

2.7 REESTRUCTURACIÓN DE CÓDIGO

El tipo más común de la reingeniería es la reestructuración del código. Algunos sistemas heredados tienen una arquitectura de programa relativamente sólida, pero los módulos individuales han sido codificados de una forma que hace difícil comprenderlos, comprobarlos y mantenerlos. En estos casos, se puede reestructurar el código ubicado dentro de los módulos sospechosos.

Para llevar a cabo esta actividad, se analiza el código fuente mediante una herramienta de reestructuración, se indican las violaciones de las estructuras de programación estructurada, y entonces se reestructura el código. El código reestructurado resultante se revisa, se comprueba para asegurarse de que no se hayan introducido anomalías y se actualiza la documentación interna del código.

La reestructuración del código se realiza para generar un diseño que produzca la misma función que el programa original, pero con mayor calidad. En general, las técnicas de reestructuración de código modelan la lógica del programa utilizando álgebra booleana y luego aplican una serie de reglas de transformación que producen lógica reestructurada. El objetivo es tomar el código fuente original y convertirlo en un diseño de procedimiento que concuerde con la filosofía de la programación estructurada.

También se utilizan otras técnicas de reestructuración para utilizarlas con las herramientas de reingeniería. Un diagrama de intercambio de recursos correlaciona cada módulo de programa y los recursos (tipos de datos, procedimientos y variables) que se intercambian entre ellos y otros módulos. Mediante la creación de representaciones del flujo de recursos se puede reestructurar la arquitectura del programa para lograr mínimos acoplamientos entre módulos.

2.8 REESTRUCTURACIÓN DE DATOS

Un programa que posea una estructura de datos débil será difícil de adaptar y de mejorar. De hecho, para muchas aplicaciones, la arquitectura de datos tiene más que ver con la viabilidad a largo plazo del programa que el propio código fuente.

A diferencia de la reestructuración de código, que se produce en un nivel relativamente bajo de abstracción, la estructuración de datos es una actividad de reingeniería a gran escala. En la mayoría de los casos, la reestructuración de datos comienza por una reingeniería inversa. La arquitectura de datos actual se analiza minuciosamente y se definen los modelos de datos necesarios. Se identifican los objetos de datos y atributos, y a continuación, se revisan las estructuras de datos a efectos de calidad.

Cuando la estructura de datos es débil (por ejemplo, actualmente se implementan archivos planos, cuando un enfoque relacional simplificaría muchísimo el procesamiento), se aplica una reingeniería a los datos.

Dado de que la reingeniería de los datos tiene una gran influencia sobre la arquitectura del programa, y también sobre los algoritmos que lo conforman, los cambios en los datos darán lugar invariablemente a cambios o bien de arquitectura o bien de código.

La ingeniería inversa de datos ocurre en diferentes grados de abstracción y con frecuencia es la primera tarea de reingeniería. Al nivel del programa, las estructuras de datos internos del programa usualmente deben someterse a reingeniería inversa como parte de un esfuerzo global de reingeniería. En el nivel del sistema, las estructuras globales de datos (por ejemplo, archivos, bases de datos) con frecuencia se someten a reingeniería para ajustarlos con los nuevos paradigmas de gestión de bases de datos (por ejemplo, el movimiento de los archivos planos hacia los sistemas de bases de datos relacionales u orientados a objetos). La ingeniería inversa de las actuales estructuras globales de datos establece el escenario para la introducción de una nueva base de datos que abarque todo el sistema.

Estructura de datos internos. Las técnicas de ingeniería inversa para datos internos del programa se enfocan en la definición de clases de objetos. Esto se logra al examinar el código del programa con el propósito de agrupar las variables de programa relacionadas. En muchos casos, la organización de los datos dentro del código identifica tipos abstractos de datos. Por ejemplo, estructuras de registro, archivos, listas y otras estructuras de datos con frecuencia ofrecen una indicación inicial de las clases.

Estructura de bases de datos. Sin importar su organización lógica y estructura física, una base de datos permite la definición de objetos de datos y apoya algún método para establecer relaciones entre los objetos. En consecuencia, la reingeniería de un esquema de bases de datos requiere comprender los objetos existentes y sus relaciones.

Los siguientes pasos se pueden utilizar para definir el modelo de datos existente como un precursor para la reingeniería de un nuevo modelo de base de datos: 1) construcción de un modelo inicial de objeto, 2) determinación de los candidatos clave, 3) refinar las clases tentativas, 4) definición de generalizaciones y 5) descubrimiento de asociaciones. Una vez que se conoce la información definida en los pasos precedentes, se aplica una serie de transformaciones para correlacionar la estructura antigua de la base de datos con una nueva estructura de base de datos. Como mínimo necesario la base de datos deberá encontrarse en la tercera forma normal.

2.9 INGENIERÍA DEL DISEÑO

La meta de la ingeniería del diseño es producir un modelo de representación que muestre firmeza, comodidad y placer. Para lograrlo, se debe practicar la diversificación y después la convergencia. La diversificación es la adquisición de un repertorio de alternativas, la materia prima del diseño: componentes, soluciones de componentes y conocimiento, todo conocimiento contenido en catálogos, libros de texto y en la mente. Una vez que se ha integrado este conjunto de información, se debe elegir y tomar elementos del repertorio que cumplan los requisitos definidos por la ingeniería de requisitos. Cuando esto ocurre se consideran y se rechazan las alternativas y se converge en una configuración particular de componentes y, por lo tanto, en la creación del producto final.

La diversificación y la convergencia demandan intuición y juicio, cualidades que están basadas en la experiencia de construir entidades similares, un conjunto de principios que guían como evoluciona el modelo, un conjunto de criterios que permiten juzgar la calidad y un proceso de iteración que conduce a una representación del diseño final.

Existe un conjunto de conceptos fundamentales de diseño de software. Cada uno ofrece un fundamento sobre el cual pueden aplicarse métodos de diseño más elaborados. Los conceptos fundamentales del diseño de software ofrecen el marco de trabajo necesario para hacer las cosas “del modo correcto”:

Abstracción. Cuando se tiene en consideración una solución modular a cualquier problema, se pueden exponer muchos niveles de abstracción. En el nivel más alto (superficial) de abstracción, la solución se pone como una medida extensa empleando el lenguaje del entorno del problema. En niveles inferiores de abstracción, se toma una orientación más procedimental. Finalmente, en el nivel más bajo (profundo) de abstracción, se establece la solución para poder implementarse directamente.

Cada paso del proceso del software es un refinamiento en el nivel de abstracción de la solución del software. Durante la ingeniería del sistema, el software se asigna como un elemento de un sistema basado en computadora. Durante el análisis de los requisitos del software, la solución del software se establece en estos términos: “aquellos que son familiares en el entorno del problema”. Conforme se adentra en el proceso de diseño, se reduce el nivel de abstracción y el nivel más bajo se alcanza cuando se genera el código fuente. A medida que se adentra en diferentes niveles de abstracción, se trabaja para crear abstracciones procedimentales y de datos.

Una *abstracción procedimental* es una secuencia nombrada de instrucciones que tiene una función específica y limitada. Describe la forma o los pasos en que se debe de proceder para efectuar la función correspondiente.

Una *abstracción de datos* es una colección de datos que describe un objeto de datos a través de sus atributos.

Una *abstracción de control* es la tercera forma de abstracción que se utiliza en el diseño del software. Al igual que las abstracciones procedimentales y de datos, este tipo de abstracción implica un mecanismo de control de programa sin especificar los datos internos.

Arquitectura. La arquitectura del software es la estructura u organización de los componentes del programa (módulos), la manera en que estos componentes interactúan, y la estructura de datos que utilizan los componentes. En un sentido más amplio, sin embargo, los componentes pueden generalizarse para representar elementos importantes del sistema y sus interacciones.

Una de las metas del diseño de software es derivar una representación arquitectónica de un sistema. Esta representación sirve como el marco de trabajo a partir del cual se conducen actividades de diseño más detalladas. Un conjunto de patrones arquitectónicos permite que se puedan reutilizar conceptos en el nivel de diseño.

El diseño arquitectónico puede representarse al usar uno o más de muchos modelos diferentes. Los *modelos estructurales* representan la arquitectura como una colección organizada de componentes del programa. Los modelos del marco de trabajo incrementan el grado de abstracción del diseño al intentar identificar marcos de trabajo repetibles del diseño arquitectónico que se encuentran en tipos de aplicaciones similares. Los *modelos dinámicos* los abordan los aspectos conductuales de la arquitectura del programa, al indicar cómo puede cambiar la configuración de la estructura o el sistema, como función de los eventos externos. Los *modelos del proceso* se centran en el diseño del proceso técnico o de negocios que el sistema debe contener. Por último, los *modelos funcionales* pueden utilizarse para representar la jerarquía funcional de un sistema.

Patrones. Un patrón de diseño es una semilla de conocimiento, la cual tiene un nombre y transporta la esencia de una solución probada a un problema recurrente dentro de cierto contexto en medio de intereses en competencia.

La finalidad de cada patrón de diseño es proporcionar una descripción que permita determinar 1) si el patrón es aplicable al trabajo actual, 2) si el patrón se puede reutilizar (por ende, ahorrar tiempo del diseño), y 3) si el patrón puede servir como guía para desarrollar un patrón similar, pero diferente en cuanto a la funcionalidad o estructura.

Modularidad. Los patrones de arquitectura y diseño de software materializan la *modularidad*; es decir, el software se divide en componentes con nombres independientes y que es posible abordar de manera individual. Estos componentes llamados *módulos* se integran para satisfacer los requisitos del problema.

Se ha establecido que la “modularidad es el atributo particular del software que permite que un programa sea manejable de manera intelectual”. El software monolítico (es decir, un programa grande compuesto por un único módulo) no puede ser entendido fácilmente debido a la cantidad de rutas de control, la amplitud de referencias, la cantidad de variables y la complejidad global hará que el entendimiento sea muy difícil. Esto conduce a una estrategia de “divide y vencerás”. Es más fácil resolver un problema complejo cuando éste se divide en piezas manejables. De hecho es un argumento para la modularidad.

Es posible concluir que si el software se subdivide, el esfuerzo requerido para desarrollarlo se reducirá en forma sensible, pero se debe tener cuidado de evitar la modularidad excesiva o insuficiente.

Un diseño y el programa resultante se modularizan de manera que el desarrollo se pueda planear con mayor facilidad; se puedan definir y entregar incrementos del software; los cambios puedan ajustarse con mayor facilidad; las pruebas y la eliminación de errores se puedan conducir con mayor eficiencia, y el mantenimiento se pueda realizar sin efectos laterales de consideración.

Ocultación de la información. Los módulos deben especificarse y diseñarse de manera que la información (procedimiento y datos) que está dentro del módulo sea inaccesible para otros módulos que no necesiten esa información.

La ocultación implica que se puede conseguir una modularidad efectiva al definir una modularidad efectiva al definir un conjunto de módulos independientes que se comuniquen entre sí y que intercambien sólo la información necesaria para lograr la función del software. La abstracción ayuda a definir las entidades de procedimiento (o información) que conforman el software. La ocultación define y fortalece las restricciones de acceso para los detalles del procedimiento dentro de un módulo y para cualquier estructura de datos local que utilice el módulo.

El uso de la ocultación de información, como un criterio de diseño para sistemas modulares, proporciona los mayores beneficios cuando se requieren modificaciones durante la realización de las pruebas y, después, en el curso de mantenimiento del software. Como la mayoría de los datos y procedimientos está oculta de las otras partes del software, existe una probabilidad menor de introducir errores inadvertidos al realizar las modificaciones y propagarlos a otros lugares dentro del software.

Independencia funcional. La independencia funcional es la suma de la modularidad y de los conceptos de abstracción y ocultación de la información.

La independencia funcional se consigue al desarrollar módulos con una función “determinante” y una “aversión” a la interacción excesiva con otros módulos. Dicho de otra manera, se desea diseñar el software de tal manera que cada módulo aborde una subfunción específica de los requisitos y tenga una sola interfaz cuando se observe desde otras partes de la estructura del programa.

El software con módulos independientes es más fácil de desarrollar porque la función se puede fraccionar y las interfaces se simplifican (y es considerable cuando el software se desarrolla en equipo). Los módulos independientes son más fáciles de mantener (y probar) porque se limitan los efectos secundarios que originan las modificaciones al diseño o al código, se reduce la propagación de errores, y es posible emplear módulos reutilizables. La independencia funcional es una clave para el buen diseño, y el diseño es la clave para lograr la calidad del software.

La independencia se evalúa aplicando dos criterios cualitativos: cohesión y acoplamiento. La *cohesión* es una medida de la fuerza funcional relativa de un módulo. El *acoplamiento* es una medida de la interdependencia relativa entre los módulos.

La cohesión es una extensión natural del concepto de ocultación de información. Un módulo cohesivo realiza una sola tarea, para lo que requiere muy poca interacción con otros componentes en otras partes del programa. O sea, debe de hacer sólo una cosa.

El acoplamiento es una medida de la interconexión entre los módulos de una estructura de software. El acoplamiento depende de la complejidad de la interfaz entre los módulos, el punto donde se realiza una entrada o referencia a un módulo, y los datos que pasan a través de la interfaz. En el diseño de software se intenta conseguir el acoplamiento más bajo posible. Una conectividad sencilla entre los módulos da como resultado un software más fácil de entender y menos propenso a experimentar el "efecto ola", el cual se presenta cuando surgen problemas en un lugar y después se propagan a través del sistema.

Refinamiento. El *refinamiento* paso a paso es una estrategia de diseño descendente que propuso inicialmente Niklaus Wirth. El desarrollo de un programa se realiza al refinar de manera sucesiva los niveles de detalle procedimentales. Una jerarquía se desarrolla al descomponer el enunciado macroscópico de una función (una abstracción procedimental) paso a paso hasta alcanzar las oraciones del lenguaje de programación.

El refinamiento es un proceso de *elaboración*. Se inicia con el enunciado de una función (o una descripción de datos) que se define con un alto grado de abstracción. Esto es, el enunciado describe los datos o la función de manera conceptual, pero no proporciona información acerca de los trabajos internos de la función o de la estructura interna de los datos. El refinamiento hace que se trabaje sobre el enunciado original y que proporcione más y más detalles conforme se realiza cada refinamiento sucesivo (elaboración).

La abstracción y el refinamiento son conceptos complementarios. La abstracción permite especificar procedimientos y datos sin considerar detalles de grado menor. El refinamiento ayuda a revelar los detalles de grado menor mientras se realiza el diseño. Ambos conceptos auxilian en la creación de un modelo de diseño completo a medida que evoluciona la actividad de diseño.

Refabricación. Una actividad importante de diseño que sugieren muchos métodos ágiles es la refabricación, técnica de reorganización que simplifica el diseño (o código) de un componente sin cambiar su función o comportamiento. La refabricación es el proceso de cambiar un sistema de un software de tal forma que no se altere el comportamiento externo de su código (diseño) y aún así se mejore su estructura interna.

Cuando un software se refabrica el diseño existente se examina en busca de redundancias, elementos de diseño inútiles, algoritmos innecesarios o insuficientes, estructuras de datos inapropiadas o construidas de manera incorrecta, o cualquier otra falla de diseño que se pueda corregir para lograr un mejor diseño.

2.10 PRÁCTICAS DE INGENIERÍA DE SOFTWARE

La comunicación, planeación, modelado, construcción y despliegue son una serie de actividades que establecen el marco de trabajo. Todos los modelos de proceso de software pueden organizarse en este esqueleto arquitectónico. Para fines de este proyecto, se utilizará el modelo de Desarrollo Rápido de Aplicaciones, debido a que éste, es decir, la reingeniería del SITG no cae en los inconvenientes que el modelo DRA tiene.

Desarrollo Rápido de Aplicaciones (DRA). EL DRA es un modelo de proceso de software incremental que resalta un ciclo de desarrollo corto. El modelo DRA es una adaptación a "alta velocidad" del modelo en cascada en el que se logra el desarrollo rápido mediante un enfoque de construcción basado en componentes. Si se entienden bien los requisitos y se limita el ámbito del proyecto, el proceso DRA permite que un equipo de desarrollo cree un "sistema completamente funcional" dentro de un periodo muy corto (entre 60 y 90 días).

Como otros modelos de proceso, el enfoque DRA cumple con las actividades genéricas del marco de trabajo: la *comunicación* que trabaja para entender el problema de negocios y las características de información que debe de incluir el software; la *planeación* que es esencial porque varios equipos de software trabajan en paralelo sobre diferentes funciones del sistema; el *modelado* que incluye tres grandes fases (modelado de negocios, modelado de datos y modelado del proceso) y establece representaciones del diseño que sirven como base para la actividad de construcción del DRA; la *construcción* que resalta el empleo de componentes de software existentes y la aplicación de la generación automática de código; y el *despliegue* que establece una base para las iteraciones subsecuentes, si éstas son necesarias.

El modelo de proceso DRA se muestra en la figura 2.3. Se muestra que las restricciones de tiempo impuestas sobre un proyecto DRA exigen un "ámbito de escalas". Si una aplicación de negocios se puede modular de forma que cada gran función pueda completarse en menos de tres meses (mediante la aplicación del enfoque de construcción basado en componentes), ésta es una candidata para el DRA. Cada gran función se puede abordar mediante un equipo de DRA por separado, para después integrarlas y formar un todo.

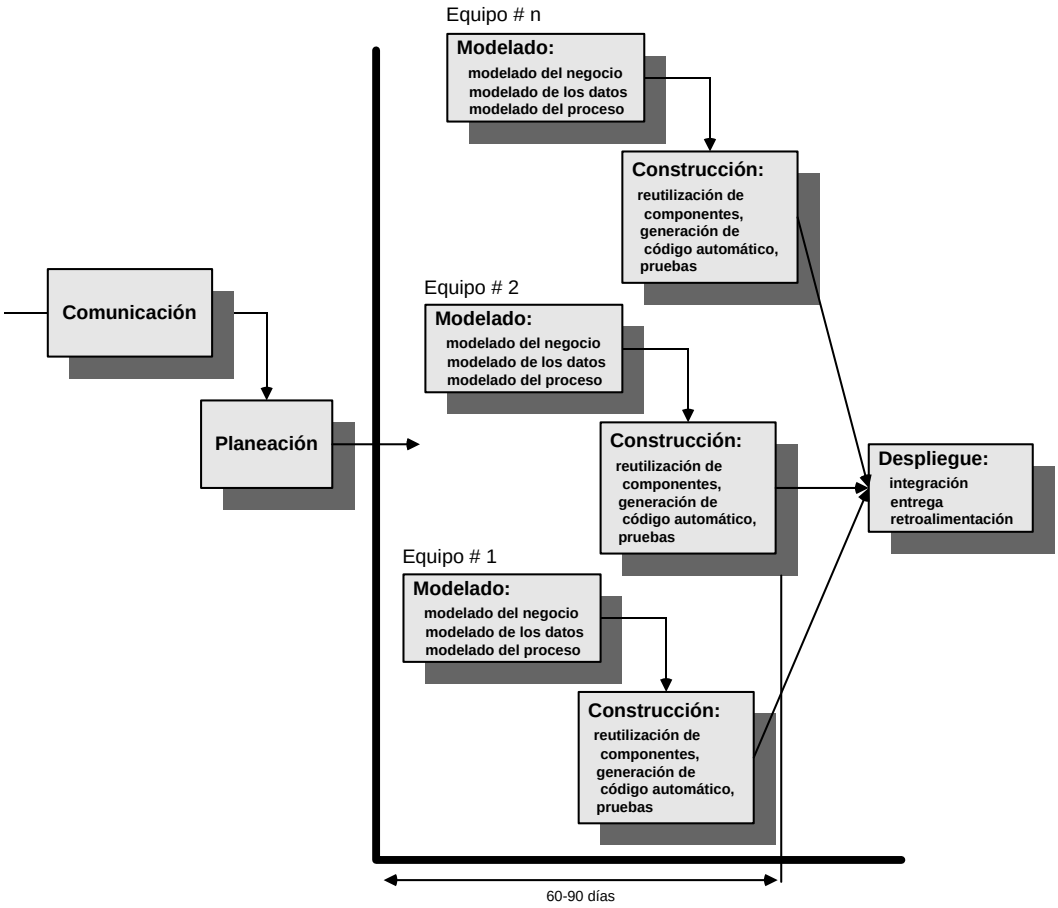


FIGURA 2.3 El modelo DRA

Como todos los modelos de proceso, el enfoque DRA tiene sus inconvenientes: 1) para proyectos grandes, pero escalables, el DRA necesita suficientes recursos humanos para crear el número correcto de equipos DRA; 2) si los desarrolladores y clientes no se comprometen con las actividades rápidas necesarias para completar el sistema en un marco de tiempo muy breve, los proyectos de DRA fallarán; 3) si un sistema no se puede modular en forma apropiada, la construcción de los componentes necesarios para el DRA será problemática; 4) si

el alto rendimiento es un aspecto importante, y se alcanzará al convertir interfaces en componentes del sistema, el enfoque DRA podría no funcionar; y 5) el DRA sería inapropiado cuando los riesgos técnicos son altos (por ejemplo, cuando una aplicación nueva aplica muchas nuevas tecnologías).

3 DEFINICIÓN DEL NEGOCIO

La Tesorería General de la Asamblea Legislativa del Distrito Federal, dependiente de la Comisión de Gobierno, tiene a su cargo la administración del presupuesto de la Asamblea, entregar las dietas de los Diputados, cubrir los sueldos y demás remuneraciones a los servidores públicos y empleados de la Asamblea, así como realizar los descuentos de carácter legal que se le ordenen; velar por el adecuado control y la exacta aplicación de los recursos presupuestales que sean proporcionados a la Asamblea para cubrir sus gastos de operación; rendir cuentas al Comité de Administración respecto del ejercicio presupuestal a su cargo e intervenir en los actos y contratos en los que la Asamblea sea parte y cuya celebración suponga una afectación directa al presupuesto de egresos de la propia Asamblea, entre otras. Para su funcionamiento y operación la Tesorería General cuenta con las siguientes áreas que la conforman:

Dirección General de Presupuesto. Tiene a su cargo ejecutar los procesos de programación, presupuestación, control y registro del ejercicio del gasto, a fin de optimizar la utilización de los recursos financieros asignados y proporcionar a los órganos de control y evaluación, información veraz y oportuna para la toma de decisiones.

Entre sus funciones tiene: Integrar el anteproyecto de programa-presupuesto de egresos de la Asamblea Legislativa y la propuesta de calendario de ejercicio correspondiente, para someterlo a la consideración de la Tesorería General; Integrar, analizar y evaluar las propuestas de adecuación programática-presupuestal para someterlas a la consideración de la Tesorería General y, en su caso aplicar y registrar las adecuaciones autorizadas; Elaboración de las cuentas por pagar de la Asamblea Legislativa, previo análisis de la suficiencia presupuestal y de la documentación comprobatoria del gasto presentada por las unidades ejecutoras del mismo, vigilando que el ejercicio presupuestal se ajuste al calendario autorizado; El registro presupuestal en el sistema integral de la Tesorería General las cuentas por pagar emitidas; Codificar, registrar y contabilizar la información relativa a las operaciones financieras derivadas de las actividades desarrolladas en la Asamblea Legislativa, verificando que se cuente con la documentación fuente requerida, conforme a la normatividad presupuestal, contable, fiscal y de control interno establecida; La realización y conciliación de las cuentas presupuestales, financieras y contables institucionales; La integración de la información presupuestal, contable y financiera requerida para su incorporación en la cuenta pública de la Asamblea Legislativa para su presentación a las instancias de control interno y externo; y La elaboración de los estados financieros de la Asamblea Legislativa, así como la información complementaria, reportes y análisis contables, presupuestales y financieros correspondientes.

Dirección General de Pagos. Tiene a su cargo administrar el flujo de efectivo de la Asamblea Legislativa del Distrito Federal, vigilando que su ejercicio se realice conforme al presupuesto autorizado, a la normatividad establecida y con base en políticas de racionalidad, disciplina y austeridad que le instruyan las instancias de gobierno.

Entre sus funciones tiene: Dar cumplimiento, en forma oportuna, a las obligaciones fiscales de la Asamblea Legislativa, así como a las asumidas ante entidades u organismos de seguridad social y terceros por conceptos de previsión social; Realizar el pago oportuno de los compromisos y obligaciones financieras que se derivan de las actividades desarrolladas en la

Asamblea Legislativa, vigilando que todo egreso cumpla con los requerimientos de control interno establecidos; Formular, documentar y realizar el pago de dietas, asignaciones, sueldos y demás conceptos de retribución y prerrogativas a Diputados, Grupos Parlamentarios y Servidores Públicos de la Asamblea Legislativa del Distrito Federal, aplicando los acuerdos emitidos por el órgano de gobierno y los movimientos e incidencias emitidas por la Oficialía Mayor; así como, calcular y registrar las retenciones de las obligaciones fiscales, aportaciones de seguridad social y demás descuentos correspondientes; Coordinar y supervisar el trámite de envío de la información y documentación de las operaciones financieras realizadas en la Dirección General de Pagos a la Dirección General de Presupuesto; Proponer el presupuesto relativo a servicios personales conforme a las plazas autorizadas por el órgano de gobierno y los techos presupuestales para prestadores de servicios asimilados a salarios que se aprueben para tal efecto en las instancias de gobierno; Coordinar el pago de las obligaciones derivadas de los contratos de arrendamiento para la instalación y operación de los Módulos de Atención, Orientación y Quejas Ciudadanas; y Coordinar y supervisar la conciliación mensual de las operaciones financieras de ingreso y egreso institucional, verificando la exactitud de los movimientos registrados.

4 INGENIERÍA DE REQUISITOS

En la Dirección General de Pagos es en donde existe la mayoría de los requisitos que hay que mejorar o implementar en el sistema, tanto para el área de Nóminas, como para el área de Pagos además de que mucha de la información de estas dos áreas alimentan a las áreas de presupuesto y contabilidad de la Dirección General de Presupuesto.

A continuación se detallan los requerimientos que se tienen por cada área de ambas Direcciones Generales:

Nómina:

- Cálculo del ISR de acuerdo a la Ley de Impuestos Sobre la Renta vigente.
- Kardex de empleados para registrarlos con un solo número y saber si cuando ingresan a la Asamblea se trata de un reingreso o una alta nueva y tener todo su historial de puestos, áreas de adscripción y fechas en las que estuvo activo.
- Cálculos para el ISSSTE: Seguro de Retiro, Retiro/Cesantía/Edad avanzada/Vejez y Cuota ISSSTE.
- Cálculo de pagos por concepto de préstamos hipotecarios que se efectúan al FOVISSSTE.
- Cálculo de Tiempo Extra para los empleados de base desglosando parte exenta y parte gravada que entra al cálculo del ISR, así como guardar el historial de estos cálculos para futuras aclaraciones a los empleados que lo soliciten.
- Llevar un registro de los préstamos Emergentes y al ISSSTE que han solicitado los empleados, así como un contador de cuotas pagadas (descuentos) para tener un mejor control de estás y sean mostradas en el recibo de pago de la nómina.
- Pagos a terceros por conceptos de descuentos como son Pensión Alimenticia, Seguro Colectivo de Retiro, Seguro de Separación Individualizada, Aportaciones a Partidos Políticos, Cuotas Sindicales de tres Sindicatos, Seguro de Vida Individual, Seguro de Gastos Médicos, Pago a las Bancas por financiamiento automotriz, Pago a Aseguradoras por Seguro automotriz y Pagos de Servicios Testamentarios.
- Pago de Servicio de Guardería para empleados de base con hijos menores de 6 años.
- Generar la impresión de los recibos de nómina en un formato preimpreso.

- Se requiere una bitácora que permita monitorear las actividades de los usuarios dentro del módulo para tener una mejor administración de los procesos que se operan.

Pagos:

- Generar la impresión de los cheques en formas preimpresas y que ahora llevarán una póliza donde se imprimirán las cuentas contables que serán afectadas en la contabilidad.
- Generar archivos planos, para envío al banco, para la protección de cheques con número de cheque y monto, de acuerdo a las especificaciones del banco.
- Generar archivos planos, para envío al banco, con la dispersión de las nóminas sólo conteniendo la relación de empleados que tienen cuenta bancaria para tal efecto, de acuerdo a las especificaciones del banco.
- Generar automáticamente las Cuentas por Pagar de las nóminas y los pagos a terceros en el módulo de Presupuesto, para poder imprimir los cheques correspondientes.

Presupuesto:

- Requiere que se generen automáticamente las Cuentas por Pagar correspondientes, de acuerdo a su origen en los módulos de Nómina y Pagos para evitar su captura manual y consecuentes errores y pérdida de tiempo.

Contabilidad:

- Requiere que se generen automáticamente las Pólizas correspondientes, de acuerdo a su origen en los módulos de Nómina, Presupuestos y Pagos para evitar su captura manual. Así mismo, se requiere incluir en cada Póliza el número de la Cuenta por Pagar que le dio origen, para poder ser analizadas más fácilmente en el reporte de Auxiliares.

5 IDENTIFICACIÓN DE PROCESOS

Los procesos se clasificaron de acuerdo a su importancia, de mayor a menor en 3 prioridades y dentro de éstas en otras subprioridades, tanto para el usuario como para las mismas actividades de la reingeniería del sistema.

Prioridad 1:

1. Cálculo del ISR.
2. Kardex de empleados.
3. Cálculos para ISSSTE.
4. Cálculos de FOVISSSTE.
5. Generar archivos para la protección de cheques.
6. Generar archivos de dispersión de nóminas.
7. Impresión de los recibos de nómina en formato preimpreso.

Prioridad 2:

1. Cálculo de Tiempo Extra.
2. Registro de faltas.
3. Registro de los préstamos con contador de cuotas pagadas (descuentos).
4. Pago de Servicio de Guardería.
5. Pagos a terceros.
6. Generar automáticamente las Cuentas por Pagar de nóminas y pagos a terceros.

Prioridad 3:

1. Creación de la bitácora de actividades del módulo de Nómina.
2. Generar la impresión de los cheques en formas preimpresas.
3. Generar automáticamente las Pólizas correspondientes de Nómina, Presupuestos y Pagos.

6 ANÁLISIS DE INVENTARIOS

A continuación se muestra el inventario de programas útiles del sistema:

Nombre	Tipo	Descripción	Programa fuente
Clases de Pago	C	Mantenimiento al catálogo de Clases de Pago (Niveles)	/sistemas/nomina/A3CLAPAG.4gl
Percepciones y Deducciones	C	Mantenimiento al catálogo de Percepciones y Deducciones	/sistemas/nomina/A3PERYDE.4gl
Adicional de Percepciones y Deducciones	C	Mantenimiento al catálogo adicional de Percepciones y Deducciones	/sistemas/nomina/A3PERYD1.4g
Centros de Costos	C	Consulta al catálogo de Centros de Costos (Presupuesto)	/sistemas/nomina/A3CENCOS.4gl
Plazas	C	Mantenimiento al catálogo de Plazas	/sistemas/nomina/A3PLAZAS.4gl
Empleados	C	Mantenimiento al catálogo de Empleados	/sistemas/nomina/A3MAEEMP.4gl
Periodos de Nómina	C	Crea los periodos de las nóminas a procesar	/sistemas/nomina/A3PERCTL.4gl
Movimientos de Percepciones y Deducciones fijos	C	Mantenimiento a los conceptos fijos de los Empleados	/sistemas/nomina/FIJOS.4gl
Cálculo de la Nómina	P	Efectúa el cálculo de la nómina	/sistemas/nomina/P3CALCRP.4g /sistemas/nomina/P3AUTOMA.4gl /sistemas/nomina/P3PASMO.4gl /sistemas/nomina/P3AUTOFI.4gl
SAR-ISSSTE	P	Traspaso de sueldo para generación de información al SAR-ISSSTE	/sistemas/nomina/sariste.4gl
Generar CXP	P	Crea la CXP de las Nóminas Canceladas	/sistemas/nomina/T3GENVUM.4gl
Traspaso CXP	P	Traspasa la CXP al Mod. Presupuesto	/sistemas/nomina/T3NOMFIN.4gl
Traspaso Cheques	P	Traspasa cheques al Mod. Presupuesto	/sistemas/nomina/T3NOMFIN2.4gl
Consulta CXP	P	Consulta las CXP del Mód de Presupuesto	/sistemas/finanzas/C1CAPVUM.4gl
Borra Cheques	P	Borra las CXP y Cheques temporales generados en el proceso Generar CXP	/sistemas/nomina/borrache.4gl
Distribución de la Nómina	R	Genera reporte a detalle de una nómina	/sistemas/nomina/R3PRENOM.4gl
Resumen de la Nómina por área	R	Genera reporte resumen de una nómina por área	/sistemas/nomina/RESUMENAREA.4gl
Resumen de la Nómina	R	Genera reporte resumen general de una nómina	/sistemas/nomina/RESUMEN.4gl
Recibos en papel Stock	R	Genera reporte de recibos de nómina para imprimir en papel Stock	/sistemas/nomina/R3RECPAP.4gl
Analítico de Percepciones	R	Genera reporte detalle de percepciones por concepto y empleado de una nómina	/sistemas/nomina/R3ANAPER.4gl
Analítico de Deducciones	R	Genera reporte detalle de deducciones por concepto y empleado de una nómina	/sistemas/nomina/R3ANADED.4gl
2% sobre Nóminas	R	Genera reporte del 2% sobre los ingresos del empleado por mes	/sistemas/nomina/DOSPORCIEN.4gl
Clases de Pago	R	Genera reporte del catálogo de Clases de Pago (niveles)	/sistemas/nomina/R3CLAPAG.4gl
Percepciones y Deducciones	R	Genera reporte del catálogo de Percepciones y Deducciones	/sistemas/nomina/R3PERYDE.4gl
Plazas	R	Resumen de Plazas ocupadas por nivel	/sistemas/nomina/R3PLAZAS.4gl
Empleados	R	Reporte del Catálogo de Empleados	/sistemas/nomina/R3MAEEMP.4gl

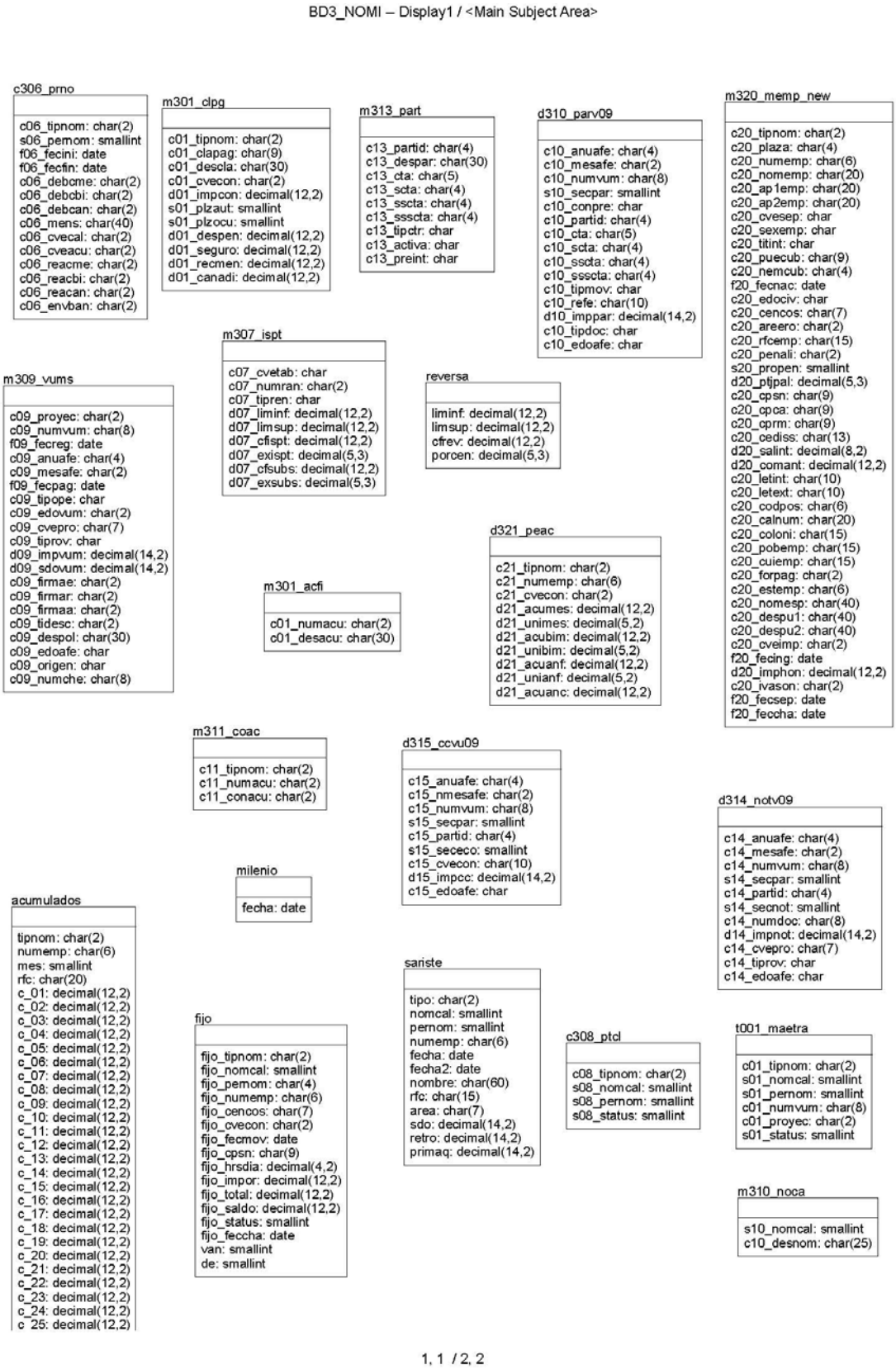
Nombre	Tipo	Descripción	Programa fuente
Parámetros Generales	C	Mantenimiento a los parámetros generales del sistema por tipo de nómina	/sistemas/nomina/A3PARSIS.4gl
Otros Parámetros Generales	C	Mantenimiento a otros parámetros generales del sistema por tipo de nómina	/sistemas/nomina/A3PARSI1.4gl
Super Bajas	P	Mantenimiento a conceptos de una nómina ya calculada	/sistemas/nomina/SUPERBAJA.4gl
Nóminas de Cálculo	C	Mantenimiento al catálogo de nóminas de cálculo	/sistemas/nomina/A3NOMCAL.4gl
Inicialización de PyD	P	Inicialización de una nómina de Dieta u Honorarios	/sistemas/nomina/A3INIPYD.4gl
Inicializa Estructura	P	Inicialización de una nómina de Estructura, Base y Confianza	/sistemas/nomina/A3INIMEN.4gl
ISSSTE	P	Generación archivo ISSSTE	/sistemas/nomina/ENVISSSTE.4gl
FOVISSSTE	P	Generación archivo FOVISSSTE	/sistemas/nomina/ENVFOVISTE.4gl
Asegurados	P	Generación archivo de Registro de Asegurados	/sistemas/nomina/ASEGURADO1.4gl
Tabla ISPT	C	Mantenimiento a la tabla de ISPT	/sistemas/nomina/A3TAISPT.4gl
Crédito al Salario	C	Mantenimiento a la tabla de Crédito al Salario	/sistemas/nomina/crealsal.4gl
CONSAR	P	Genera archivo para la CONSAR	/sistemas/nomina/consar.4gl
Carga de Presupuestos	C	Registrar el presupuesto anual asignado	/sistemas/finanzas/C1PTOCEC.4gl /sistemas/finanzas/C1PTOPRG.4gl /sistemas/finanzas/C1PTOPAR.4gl
Ampliaciones de Presupuestos	C	Registrar una ampliación al presupuesto	/sistemas/finanzas/T1AMPLCC.4gl /sistemas/finanzas/T1AMPLPR.4gl /sistemas/finanzas/T1AMPLPA.4gl
Reducciones de Presupuestos	C	Registrar una reducción al presupuesto	/sistemas/finanzas/T1REDUCC.4gl /sistemas/finanzas/T1REDUPR.4gl /sistemas/finanzas/T1REDUPA.4gl
Transferencias de Presupuestos	C	Registrar una transferencia del presupuesto	/sistemas/finanzas/T1TRANCC.4gl /sistemas/finanzas/T1TRANPR.4gl /sistemas/finanzas/T1TRANPA.4gl
Traspasos de Presupuestos	C	Registrar un traspaso en el presupuesto	/sistemas/finanzas/T1TRASPR.4gl /sistemas/finanzas/T1TRASPA.4gl
Solicitud de Cheque	P	Hacer una cheque por solicitud	/sistemas/finanzas/A1SOLCHE.4gl
Cuenta X Pagar	P	Registrar una Cuenta por Pagar	/sistemas/finanzas/C1CAPVUM.4gl
Cambios de Centros de Costos	P	Realiza cambios de Centro de Costos a una CXP	/sistemas/finanzas/A1CAMCC.4gl
Cambios de Notas	P	Realiza cambios de Notas a una CXP	/sistemas/finanzas/A1CAMNOT.4gl
Selecciona CXP	P	Seleccionar las CXP que están listas para pagarse y/o afectarse	/sistemas/finanzas/T1SELPAG.4gl
Desafecta CXP	P	Desafectar presupuestalmente una CXP	/sistemas/finanzas/DESAFECTA.4gl
Selección de Cheques	P	Selecciona los cheques que se imprimirán	/sistemas/finanzas/A1EMICHE.4gl
Modificar Cheques	P	Modifica el beneficiario y/o los conceptos de un cheque	/sistemas/finanzas/A1MODCHE.4gl
Impresión de Cheques	R	Genera la impresión de los cheques que ya fueron seleccionados	/sistemas/finanzas/A1IMPCHE.4gl
Impresión Bancomer	R	Imprime cheques Bancomer de 3 por hoja	/sistemas/finanzas/CHEQUE20.4gl
Impresión Emis	R	Información de queches en emis	/sistemas/finanzas/A1CHEREI.4gl
Cancelación Cheque Desafectación	P	Realiza la cancelación de cheques y desafectación	/sistemas/finanzas/A1CHECAN.4gl
Cancela Nómina	P	Realiza la cancelación de cheques de Nómina y desafecta	/sistemas/finanzas/A1CANNOM.4gl
Generar Cheques de Nómina	P	Prepara la información de Nómina para obtener el formato de los cheques	/sistemas/nomina/T3GENVUM.4gl
Impresión de Cheques de Nómina	R	Permite imprimir lo cheques que se encuentran generados	/sistemas/nomina/A3IMPCHE.4gl
Reimpresión de Cheques de Nómina	R	Permite reimprimir lo cheques que se encuentran generados	/sistemas/nomina/A3CHEREI.4gl
Traspaso Cuenta	P	Envía la información generada de la cuenta de Nómina a Finanzas	/sistemas/nomina/T3NOMFIN.4gl
Traspaso Cheques	P	Envía la información generada de los cheques de Nómina a Finanzas	/sistemas/nomina/T3NOMFIN2.4gl

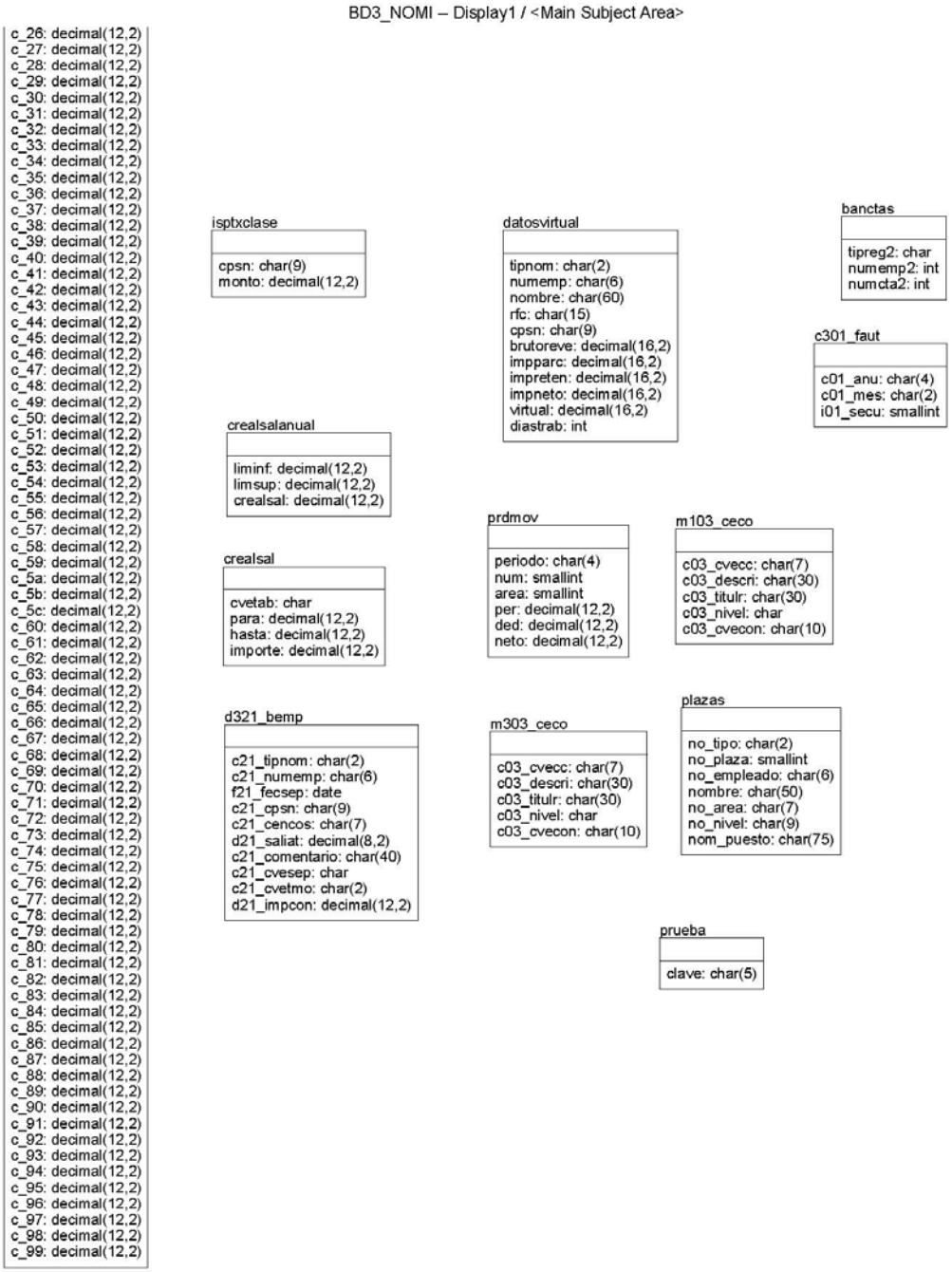
Nombre	Tipo	Descripción	Programa fuente
Borra Cheques	P	Borra la CXP y los cheques de Nómina temporales	/sistemas/nomina/borrache.4gl
Cancelación Cheques	P	Cancela un cheque	/sistemas/finanzas/T1CANCHE.4gl
Ingresos Bancos	P	Captura los ingresos Bancarios	/sistemas/finanzas/A1INGBAN.4gl
Mov Diversos	P	Captura de cargos y/o créditos varios (bancarios)	/sistemas/finanzas/A1DIVBAN.4gl
Cierre Diario	P	Realiza cierre donde se actualizan los saldos de bancos	/sistemas/finanzas/A1CIEBAN.4gl
Partidas	C	Agregar, Borrar, Modificar o Consultar Partidas	/sistemas/finanzas/A1PARTID.4gl
Centros de Costos	C	Agregar, Borrar, Modificar o Consultar Centros de Costos	/sistemas/finanzas/A1CENCOS.4gl
Programas	C	Agregar, Borrar, Modificar o Consultar Programas	/sistemas/finanzas/A1PROGRA.4gl
Rel. Partidas	C	Agregar, Borrar, Modificar o Consultar la relación de Partidas Individuales	/sistemas/finanzas/A1RELPAI.4gl
Proyectos	C	Agregar, Borrar, Modificar o Consultar Proyectos	/sistemas/finanzas/A1PROYEC.4gl
Beneficiarios	C	Agregar, Borrar, Modificar o Consultar Beneficiarios	/sistemas/finanzas/A1BENEFI.4gl
Bancos	C	Agregar, Borrar, Modificar o Consultar Bancos	/sistemas/finanzas/A1BANCOS.4gl
Firmas	C	Agregar, Borrar, Modificar o Consultar Firmas Autorizadas	/sistemas/finanzas/A1FIRMAS.4gl
Pólizas	C	Agregar, Borrar, Modificar o Consultar la descripción de Pólizas	/sistemas/finanzas/A1DESPOL.4gl
Documentos	C	Agregar, Borrar, Modificar o Consultar tipos de Documentos	/sistemas/finanzas/A1TIPDOC.4gl
Cuentas Contables Fijas	C	Agregar, Borrar, Modificar o Consultar Cuentas Contables Fijas	/sistemas/finanzas/A1CUEPAS.4gl
Estado de Cuenta	P	Alimenta los movimientos del estado de cuenta del Banco	/sistemas/finanzas/A1EDOBCO.4gl
Conciliación Automática	P	Realiza la conciliación Bancaria	/sistemas/finanzas/T1CONMOV.4gl
Reporte validación del Estado de Cuenta	R	Verifica los movimientos del Estado de Cuenta del Banco	/sistemas/finanzas/R1REPEDC.4gl
Reporte pendientes Empresa	R	Reporte de Partidas Pendientes	/sistemas/finanzas/R1BANPEN.4gl
Reporte de Cuadre	R	Carátula de la Conciliación Bancaria	/sistemas/finanzas/R1REPCUA.4gl
Reporte Cargos-Créditos Estado de Cuenta	R	Cargos y/o créditos no conciliados del Estado de Cuenta	/sistemas/finanzas/R1COCREC.4gl
Reporte Cargos-Créditos	R	Cargos y/o créditos no conciliados	/sistemas/finanzas/R1COCRBA.4gl
Presupuestos	R	Genera reporte por Partidas, Conceptos o Capítulos	/sistemas/finanzas/R1REPPTO.4gl
Calendario Programa	R	Genera reporte Calendarizado por Programa del Presupuesto Asignado	/sistemas/finanzas/R1REPPRGCAL.4gl
Calendario Programa Modificado	R	Genera reporte Calendarizado por Programa del Presupuesto Modificado	/sistemas/finanzas/repprgcalmod.4gl
Verifica Presupuesto	R	Verificación del Presupuesto de la captura inicial	/sistemas/finanzas/R1REPCEC.4gl /sistemas/finanzas/R1REPPRG.4gl /sistemas/finanzas/R1REPPAR.4gl
Movimientos al Presupuesto	R	Reporte de Movimientos al Presupuesto Ampliación/Reducción	/sistemas/finanzas/R1MOVPTO.4gl
Desglose de Documentos (todas)	R	Genera reporte de Cuenta Comprobada	/sistemas/finanzas/R1CUECOM.4gl
Desglose de Documentos (una)	R	Genera reporte de Cuenta Comprobada	/sistemas/finanzas/R1CUECOM1.4gl
Economía Cuenta Comprobada	R	Selección de reporte de Economía de Cuenta Comprobada	/sistemas/finanzas/R1REPPTO_cc.4gl
Partidas	R	Genera reporte por Partidas	/sistemas/finanzas/R1PARTID.4gl
Centros de Costos	R	Genera reporte por Centros de Costos	/sistemas/finanzas/R1CENCOS.4gl
Relación Partidas	R	Genera reporte de relación de Partidas Individuales	/sistemas/finanzas/R1RELPAR.4gl
Proyectos	R	Genera reporte de Proyectos	/sistemas/finanzas/R1PROYEC.4gl
Beneficiarios Numérico	R	Genera reporte de Beneficiarios Numérico	/sistemas/finanzas/R1BENEFI1.4gl
Beneficiarios Alfabético	R	Genera reporte de Beneficiarios Alfabético	/sistemas/finanzas/R1BENEFI.4gl
Bancos	R	Genera reporte de Bancos	/sistemas/finanzas/R1BANCOS.4gl
Firmas	R	Genera reporte de Firmas Autorizadas	/sistemas/finanzas/R1FIRMAS.4gl
Pólizas	R	Genera reporte de Descripción de Pólizas	/sistemas/finanzas/R1DESPOL.4gl

Nombre	Tipo	Descripción	Programa fuente
Documentos	R	Genera reporte de Tipos de Documentos	/sistemas/finanzas/R1TIPDOC.4gl
Cuentas Contables	R	Genera reporte de Cuentas Contables	/sistemas/finanzas/R1CUEPAS.4gl
Programas	R	Genera reporte de Programas	/sistemas/finanzas/R1PROGRA.4gl
Cheques	R	Obtención de la relación de cheques por clave de vigencia (status)	/sistemas/finanzas/R1REPCHE.4gl
Bancos	R	Relación de movimientos de chequeras (ingresos, egresos)	/sistemas/finanzas/R1EDOBAN.4gl
Bancos 2	R	Relación de movimientos de chequeras (sin totales)	/sistemas/finanzas/R1EDOBAN2.4gl
Bancos 3	R	Relación de movimientos de chequeras (ingresos, egresos sin descripciones)	/sistemas/finanzas/R1EDOBAN3.4gl
Cuenta por Pagar	R	Genera el reporte de una Cuenta por Pagar con su desglose completo	/sistemas/finanzas/R1CAPVUM.4gl
Resumen Cuenta por Pagar	R	Muestra un análisis de todas las Cuentas por Pagar	/sistemas/finanzas/R1EDOCXP.4gl
Saldo Bancario	R	Presenta el Saldo Bancario de un fecha	/sistemas/finanzas/SALDO.4gl
Movimientos Bancarios por fecha y concepto	R	Presenta los movimientos Bancarios por fecha y concepto	/sistemas/finanzas/REPBAN.4gl
Resumen Partida 3503	R	Resumen de movimientos de la Partida 3503	/sistemas/finanzas/autos_3503.4gl
Detalle Partida 3503	R	Movimientos de la Partida 3503 agrupado por placas	/sistemas/finanzas/autos_3503_2.4gl
Ejercicio Mensual por Programa	R	Informe del ejercicio del Presupuesto Mensual por Programa	/sistemas/finanzas/EJEPROGMES.4gl
Partidas (una)	R	Movimientos de Partidas por mes (una)	/sistemas/finanzas/R1PARTIVUM.4gl
Partidas (todas)	R	Movimientos de Partidas por mes (todas)	/sistemas/finanzas/R1PARTIVUM1.4gl
CXP por Proveedor	R	Reporte de Cuentas por Pagar por Proveedor	/sistemas/finanzas/R1CXPBEN.4gl
Partidas por mes y CXP	R	Movimientos de Partidas por mes y Cuentas por Pagar	/sistemas/finanzas/xpartmes.4gl
Fondo Revolvente por Partidas	R	Fondos Revolventes por Partida desglosado al mes deseado	/sistemas/finanzas/fondo2.4gl
Ejercido por CC y Partida (una)	R	Ejercidos por Centro de Costos y Partida (una)	/sistemas/finanzas/R1CCPAR.4gl
Ejercido por CC y Partida (todas)	R	Ejercidos por Centro de Costos y Partida (todas)	/sistemas/finanzas/R1CCPAR2.4gl
Documentos x CC y Partida	R	Detalle de Documentos por Área, Partida y Periodo	/sistemas/finanzas/areperpar.4gl
CXP a nivel Documento	R	Reporte de Cuentas por Pagar a nivel Documento	/sistemas/finanzas/R1CXPDOC.4gl
CXP x Beneficiario a nivel Documento	R	Reporte de Cuentas por Pagar por Beneficiario a nivel Documento	/sistemas/finanzas/R1BENDOC.4gl
CXP x Partidas a nivel Documento	R	Reporte de Cuentas por Pagar por Partidas a nivel Documento	/sistemas/finanzas/R1CXPPDOC.4gl
CXP x Centro de Costos y Partidas	R	Reporte de Cuentas por Pagar por Centro de Costos y Partida	/sistemas/finanzas/R1CXPCCDOC.4gl
CXP x Partida-Beneficiario a nivel Documento	R	Reporte de Cuentas por Pagar por Partida-Beneficiario a nivel Documento	/sistemas/finanzas/R1BENDOC2.4gl
Parámetros Generales	C	Modificar los parámetros generales del sistema	/sistemas/finanzas/A1PARGRL.4gl

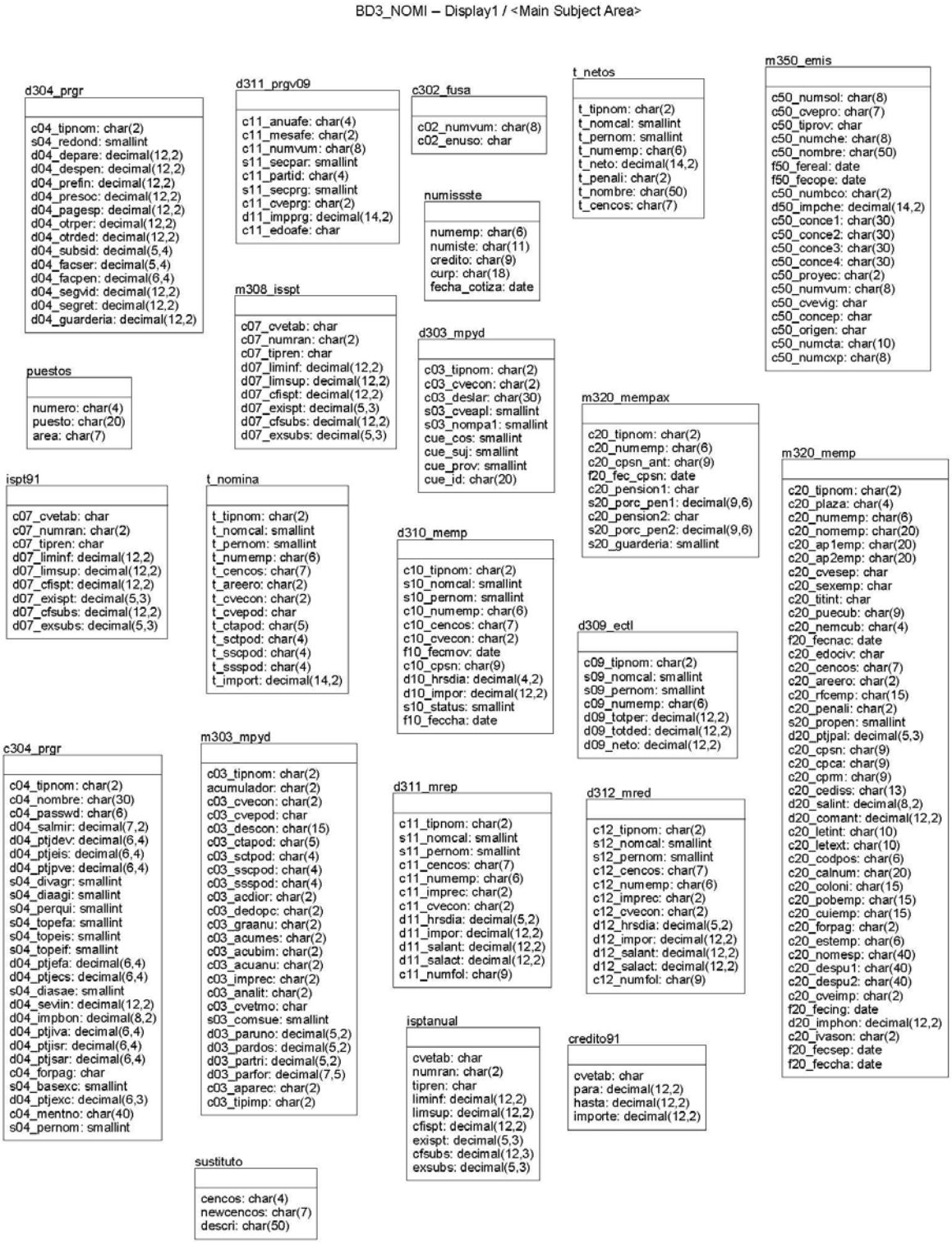
A continuación se muestran los esquemas de las bases de datos originales:

Base de datos de nómina “bd3_nomi”.

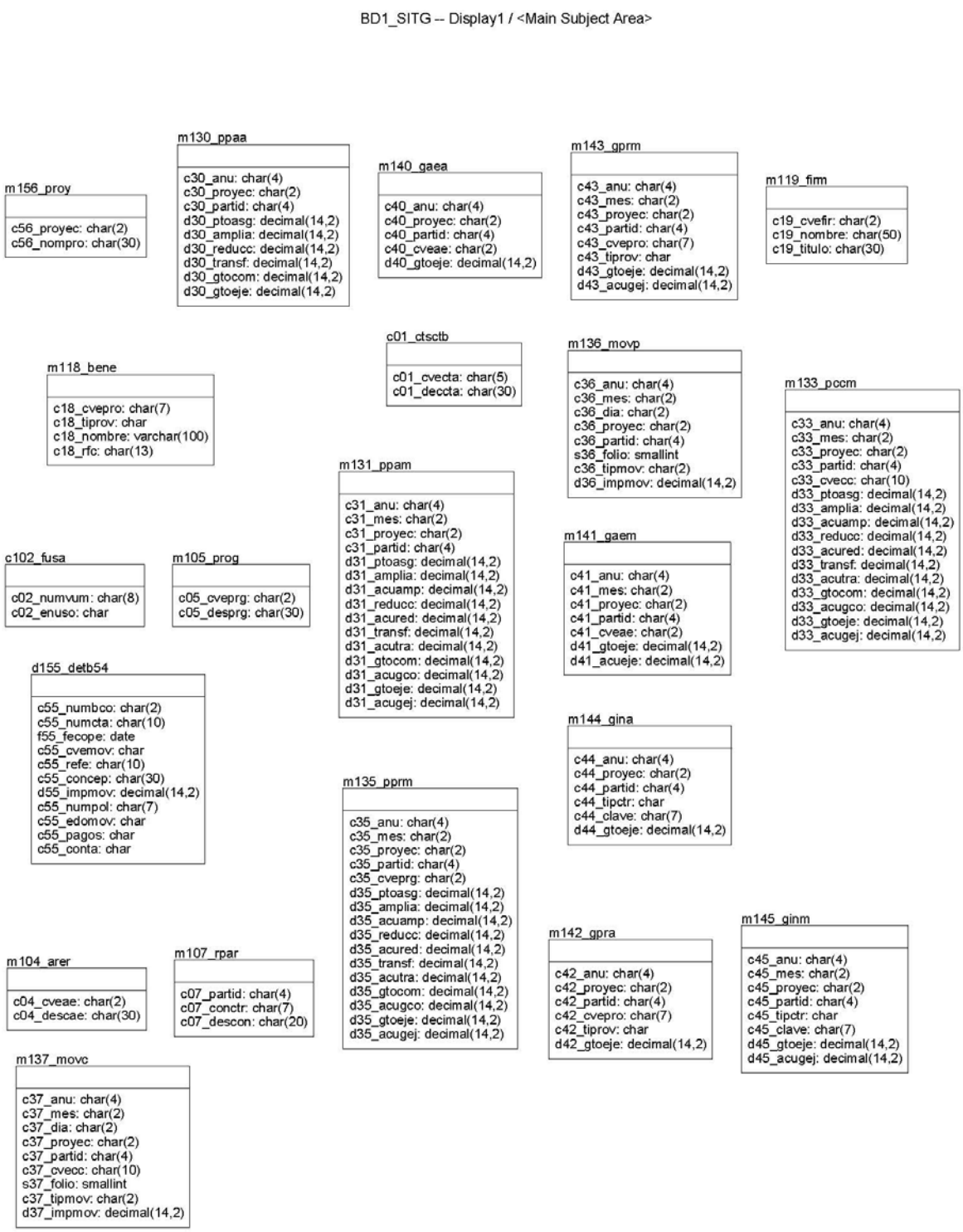


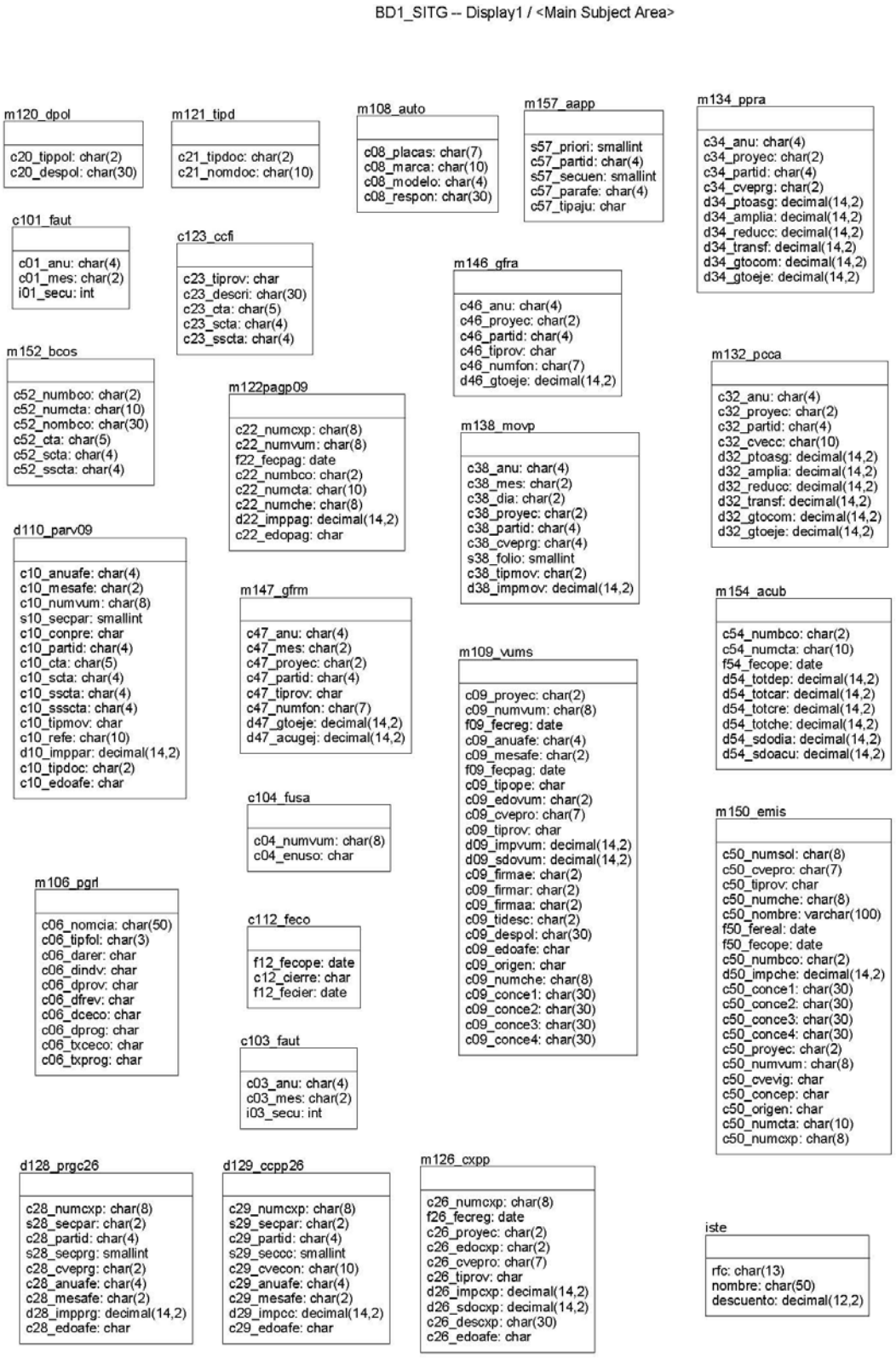

BD3_NOMI – Display1 / <Main Subject Area>
c_26: decimal(12,2)
c_27: decimal(12,2)
c_28: decimal(12,2)
c_29: decimal(12,2)
c_30: decimal(12,2)
c_31: decimal(12,2)
c_32: decimal(12,2)
c_33: decimal(12,2)
c_34: decimal(12,2)
c_35: decimal(12,2)
c_36: decimal(12,2)
c_37: decimal(12,2)
c_38: decimal(12,2)
c_39: decimal(12,2)
c_40: decimal(12,2)
c_41: decimal(12,2)
c_42: decimal(12,2)
c_43: decimal(12,2)
c_44: decimal(12,2)
c_45: decimal(12,2)
c_46: decimal(12,2)
c_47: decimal(12,2)
c_48: decimal(12,2)
c_49: decimal(12,2)
c_50: decimal(12,2)
c_51: decimal(12,2)
c_52: decimal(12,2)
c_53: decimal(12,2)
c_54: decimal(12,2)
c_55: decimal(12,2)
c_56: decimal(12,2)
c_57: decimal(12,2)
c_58: decimal(12,2)
c_59: decimal(12,2)
c_5a: decimal(12,2)
c_5b: decimal(12,2)
c_5c: decimal(12,2)
c_60: decimal(12,2)
c_61: decimal(12,2)
c_62: decimal(12,2)
c_63: decimal(12,2)
c_64: decimal(12,2)
c_65: decimal(12,2)
c_66: decimal(12,2)
c_67: decimal(12,2)
c_68: decimal(12,2)
c_69: decimal(12,2)
c_70: decimal(12,2)
c_71: decimal(12,2)
c_72: decimal(12,2)
c_73: decimal(12,2)
c_74: decimal(12,2)
c_75: decimal(12,2)
c_76: decimal(12,2)
c_77: decimal(12,2)
c_78: decimal(12,2)
c_79: decimal(12,2)
c_80: decimal(12,2)
c_81: decimal(12,2)
c_82: decimal(12,2)
c_83: decimal(12,2)
c_84: decimal(12,2)
c_85: decimal(12,2)
c_86: decimal(12,2)
c_87: decimal(12,2)
c_88: decimal(12,2)
c_89: decimal(12,2)
c_90: decimal(12,2)
c_91: decimal(12,2)
c_92: decimal(12,2)
c_93: decimal(12,2)
c_94: decimal(12,2)
c_95: decimal(12,2)
c_96: decimal(12,2)
c_97: decimal(12,2)
c_98: decimal(12,2)
c_99: decimal(12,2)
isptxclase
cpsn: char(9)
monto: decimal(12,2)
datosvirtual
tipnom: char(2)
numemp: char(6)
nombre: char(60)
rfc: char(15)
cpsn: char(9)
brutoreve: decimal(16,2)
impparc: decimal(16,2)
impreten: decimal(16,2)
impneto: decimal(16,2)
virtual: decimal(16,2)
diastrab: int
banctas
tipreg2: char
numemp2: int
numcta2: int
c301_faut
c01_anu: char(4)
c01_mes: char(2)
i01_secu: smallint
crealsalanual
liminf: decimal(12,2)
limsup: decimal(12,2)
crealsal: decimal(12,2)
prdmov
periodo: char(4)
num: smallint
area: smallint
per: decimal(12,2)
ded: decimal(12,2)
neto: decimal(12,2)
m103_ceco
c03_cvecc: char(7)
c03_descri: char(30)
c03_titulr: char(30)
c03_nivel: char
c03_cvecon: char(10)
crealsal
cvetab: char
para: decimal(12,2)
hasta: decimal(12,2)
importe: decimal(12,2)
d321_bemp
c21_tipnom: char(2)
c21_numemp: char(6)
f21_fecsep: date
c21_cpsn: char(9)
c21_cencos: char(7)
d21_saliat: decimal(8,2)
c21_comentario: char(40)
c21_cvesep: char
c21_cvetmo: char(2)
d21_impcon: decimal(12,2)
m303_ceco
c03_cvecc: char(7)
c03_descri: char(30)
c03_titulr: char(30)
c03_nivel: char
c03_cvecon: char(10)
plazas
no_tipo: char(2)
no_plaza: smallint
no_empleado: char(6)
nombre: char(50)
no_area: char(7)
no_nivel: char(9)
nom_puesto: char(75)
prueba
clave: char(5)

2, 1 / 2, 2

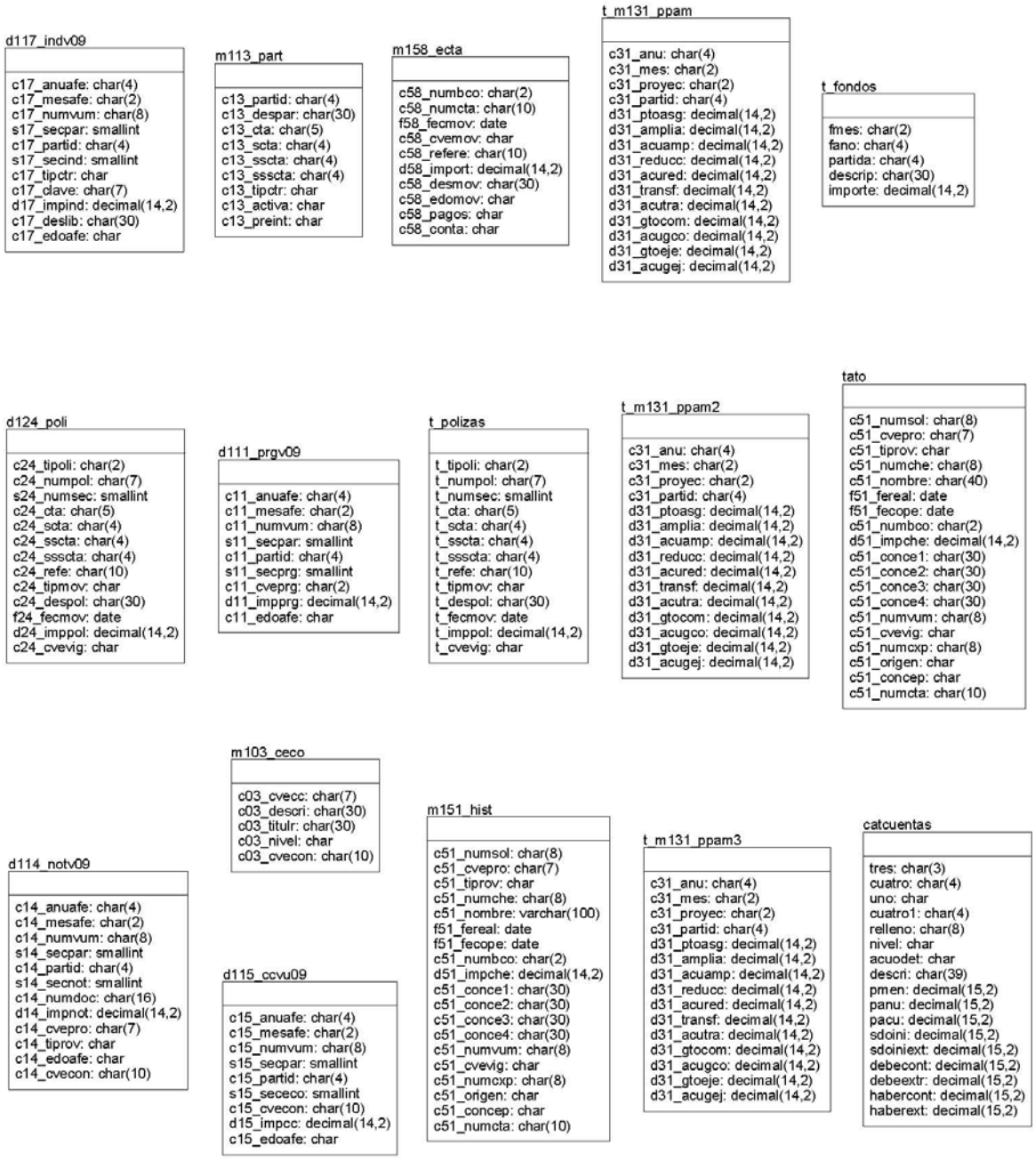

BD3_NOMI – Display1 / <Main Subject Area>
d304_prgr
c04_tipnom: char(2)
s04_redond: smallint
d04_depare: decimal(12,2)
d04_despen: decimal(12,2)
d04_prefin: decimal(12,2)
d04_presoc: decimal(12,2)
d04_pagesp: decimal(12,2)
d04_otrper: decimal(12,2)
d04_otrded: decimal(12,2)
d04_subsid: decimal(5,4)
d04_facser: decimal(5,4)
d04_facpen: decimal(6,4)
d04_segvid: decimal(12,2)
d04_segret: decimal(12,2)
d04_guarderia: decimal(12,2)
d311_prgv09
c11_anuafe: char(4)
c11_mesafe: char(2)
c11_numvum: char(8)
s11_secpar: smallint
c11_partid: char(4)
s11_secprg: smallint
c11_cveprg: char(2)
d11_impprg: decimal(14,2)
c11_edoafe: char
c302_fusa
c02_numvum: char(8)
c02_enuso: char
t_netos
t_tipnom: char(2)
t_nomcal: smallint
t_pernom: smallint
t_numemp: char(6)
t_neto: decimal(14,2)
t_penali: char(2)
t_nombre: char(50)
t_cencos: char(7)
m350_emis
c50_numsol: char(8)
c50_cvepro: char(7)
c50_tiprov: char
c50_numche: char(8)
c50_nombre: char(50)
f50_fereal: date
f50_fecope: date
c50_numbco: char(2)
d50_impche: decimal(14,2)
c50_conce1: char(30)
c50_conce2: char(30)
c50_conce3: char(30)
c50_conce4: char(30)
c50_proyec: char(2)
c50_numvum: char(8)
c50_cvevig: char
c50_concep: char
c50_origen: char
c50_numcta: char(10)
c50_numcxp: char(8)
numissste
numemp: char(6)
numiste: char(11)
credito: char(9)
curp: char(18)
fecha_cotiza: date
m308_isspt
c07_cvetab: char
c07_numran: char(2)
c07_tipren: char
d07_liminf: decimal(12,2)
d07_limsup: decimal(12,2)
d07_cfispt: decimal(12,2)
d07_exispt: decimal(5,3)
d07_cfsubs: decimal(12,2)
d07_exsubs: decimal(5,3)
d303_mpyd
c03_tipnom: char(2)
c03_cvecon: char(2)
c03_deslar: char(30)
s03_cveapl: smallint
s03_nompa1: smallint
cue_cos: smallint
cue_suj: smallint
cue_prov: smallint
cue_id: char(20)
puestos
numero: char(4)
puesto: char(20)
area: char(7)
m320_mempax
c20_tipnom: char(2)
c20_numemp: char(6)
c20_cpsn_ant: char(9)
f20_fec_cpsn: date
c20_pension1: char
s20_porc_pen1: decimal(9,6)
c20_pension2: char
s20_porc_pen2: decimal(9,6)
s20_guarderia: smallint
m320_memp
c20_tipnom: char(2)
c20_plaza: char(4)
c20_numemp: char(6)
c20_nomemp: char(20)
c20_ap1emp: char(20)
c20_ap2emp: char(20)
c20_cvesep: char
c20_sexemp: char
c20_titint: char
c20_puecub: char(9)
c20_nemcub: char(4)
f20_fecnac: date
c20_edociv: char
c20_cencos: char(7)
c20_areero: char(2)
c20_rfcemp: char(15)
c20_penali: char(2)
s20_propen: smallint
d20_ptjpal: decimal(5,3)
c20_cpsn: char(9)
c20_cpca: char(9)
c20_cprm: char(9)
c20_cediss: char(13)
d20_salint: decimal(8,2)
d20_comant: decimal(12,2)
c20_letint: char(10)
c20_letext: char(10)
c20_codpos: char(6)
c20_calnum: char(20)
c20_coloni: char(15)
c20_pobemp: char(15)
c20_cuiemp: char(15)
c20_forpag: char(2)
c20_estemp: char(6)
c20_nomesp: char(40)
c20_despu1: char(40)
c20_despu2: char(40)
c20_cveimp: char(2)
f20_fecing: date
d20_imphon: decimal(12,2)
c20_ivason: char(2)
f20_fecsep: date
f20_fecha: date
ispt91
c07_cvetab: char
c07_numran: char(2)
c07_tipren: char
d07_liminf: decimal(12,2)
d07_limsup: decimal(12,2)
d07_cfispt: decimal(12,2)
d07_exispt: decimal(5,3)
d07_cfsubs: decimal(12,2)
d07_exsubs: decimal(5,3)
t_nomina
t_tipnom: char(2)
t_nomcal: smallint
t_pernom: smallint
t_numemp: char(6)
t_cencos: char(7)
t_areero: char(2)
t_cvecon: char(2)
t_cvepod: char
t_ctapod: char(5)
t_sctpod: char(4)
t_sscpod: char(4)
t_ssspod: char(4)
t_import: decimal(14,2)
d310_memp
c10_tipnom: char(2)
s10_nomcal: smallint
s10_pernom: smallint
c10_numemp: char(6)
c10_cencos: char(7)
c10_cvecon: char(2)
f10_fecmov: date
c10_cpsn: char(9)
d10_hrsdia: decimal(4,2)
d10_impor: decimal(12,2)
s10_status: smallint
f10_feccha: date
d309_ectl
c09_tipnom: char(2)
s09_nomcal: smallint
s09_pernom: smallint
c09_numemp: char(6)
d09_totper: decimal(12,2)
d09_totded: decimal(12,2)
d09_neto: decimal(12,2)
c304_prgr
c04_tipnom: char(2)
c04_nombre: char(30)
c04_passwd: char(6)
d04_salmir: decimal(7,2)
d04_ptjdev: decimal(6,4)
d04_ptjeis: decimal(6,4)
d04_ptjpve: decimal(6,4)
s04_divagr: smallint
s04_diaagi: smallint
s04_perqui: smallint
s04_topefa: smallint
s04_topeis: smallint
s04_topeif: smallint
d04_ptjefa: decimal(6,4)
d04_ptjecs: decimal(6,4)
s04_diasae: smallint
d04_sevin: decimal(12,2)
d04_impbon: decimal(8,2)
d04_ptjiva: decimal(6,4)
d04_ptjisr: decimal(6,4)
d04_ptjsar: decimal(6,4)
c04_forpag: char
s04_basexc: smallint
d04_ptjexc: decimal(6,3)
c04_mentno: char(40)
s04_pernom: smallint
m303_mpyd
c03_tipnom: char(2)
acumulador: char(2)
c03_cvecon: char(2)
c03_cvepod: char
c03_descon: char(15)
c03_ctapod: char(5)
c03_sctpod: char(4)
c03_sscpod: char(4)
c03_ssspod: char(4)
c03_acdior: char(2)
c03_dedopc: char(2)
c03_graanu: char(2)
c03_acumes: char(2)
c03_acubim: char(2)
c03_acuanu: char(2)
c03_imprec: char(2)
c03_analit: char(2)
c03_cvetmo: char
s03_comsue: smallint
d03_paruno: decimal(5,2)
d03_pardos: decimal(5,2)
d03_partri: decimal(5,2)
d03_parfor: decimal(7,5)
c03_aparec: char(2)
c03_tipimp: char(2)
d311_mrep
c11_tipnom: char(2)
s11_nomcal: smallint
s11_pernom: smallint
c11_cencos: char(7)
c11_numemp: char(6)
c11_imprec: char(2)
c11_cvecon: char(2)
d11_hrsdia: decimal(5,2)
d11_impor: decimal(12,2)
d11_salant: decimal(12,2)
d11_salact: decimal(12,2)
c11_numfol: char(9)
d312_mred
c12_tipnom: char(2)
s12_nomcal: smallint
s12_pernom: smallint
c12_cencos: char(7)
c12_numemp: char(6)
c12_imprec: char(2)
c12_cvecon: char(2)
d12_hrsdia: decimal(5,2)
d12_impor: decimal(12,2)
d12_salant: decimal(12,2)
d12_salact: decimal(12,2)
c12_numfol: char(9)
isptanual
cvetab: char
numran: char(2)
tipren: char
liminf: decimal(12,2)
limsup: decimal(12,2)
cfispt: decimal(12,2)
exispt: decimal(5,3)
cfsubs: decimal(12,3)
exsubs: decimal(5,3)
credito91
cvetab: char
para: decimal(12,2)
hasta: decimal(12,2)
importe: decimal(12,2)
sustituto
cencos: char(4)
newcencos: char(7)
descri: char(50)
1, 2 / 2, 2

Base de datos de presupuestos “bd1_sitg” (finanzas y pagos).

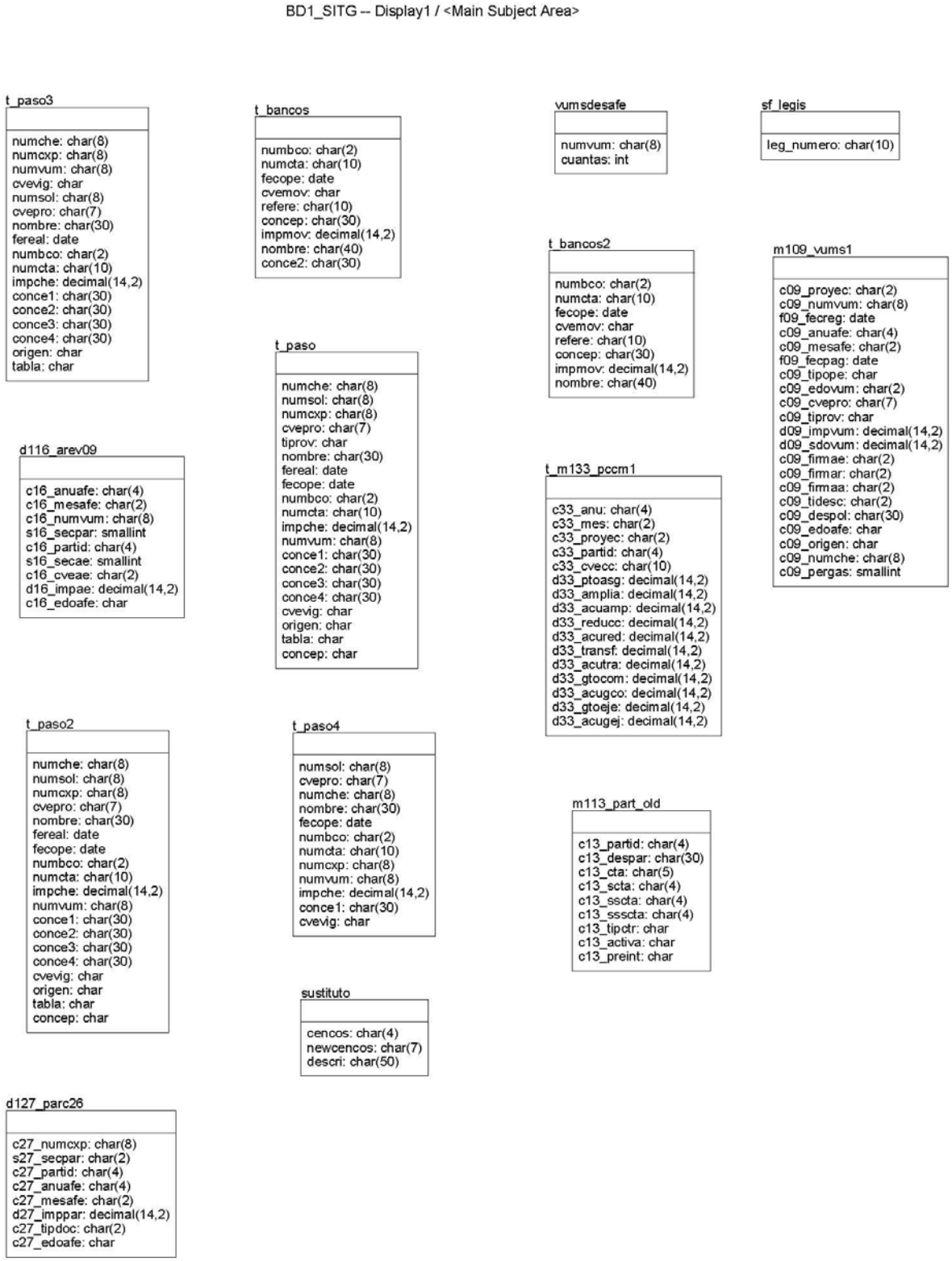



BD1_SITG -- Display1 / <Main Subject Area>
m120_dpol
c20_tippol: char(2)
c20_despol: char(30)
m121_tipd
c21_tipdoc: char(2)
c21_nomdoc: char(10)
m108_auto
c08_placas: char(7)
c08_marca: char(10)
c08_modelo: char(4)
c08_respon: char(30)
m157_aapp
s57_priori: smallint
c57_partid: char(4)
s57_secuen: smallint
c57_parafe: char(4)
c57_tipaju: char
m134_ppra
c34_anu: char(4)
c34_proyec: char(2)
c34_partid: char(4)
c34_cveprg: char(2)
d34_ptoasg: decimal(14,2)
d34_amplia: decimal(14,2)
d34_reducc: decimal(14,2)
d34_transf: decimal(14,2)
d34_gtocom: decimal(14,2)
d34_gtoeje: decimal(14,2)
c101_faut
c01_anu: char(4)
c01_mes: char(2)
i01_secu: int
c123_ccfi
c23_tiprov: char
c23_descri: char(30)
c23_cta: char(5)
c23_scta: char(4)
c23_sscta: char(4)
m146_gfra
c46_anu: char(4)
c46_proyec: char(2)
c46_partid: char(4)
c46_tiprov: char
c46_numfon: char(7)
d46_gtoeje: decimal(14,2)
m152_bcos
c52_numbco: char(2)
c52_numcta: char(10)
c52_nombco: char(30)
c52_cta: char(5)
c52_scta: char(4)
c52_sscta: char(4)
m122pagp09
c22_numcxp: char(8)
c22_numvum: char(8)
f22_fecpag: date
c22_numbco: char(2)
c22_numcta: char(10)
c22_numche: char(8)
d22_imppag: decimal(14,2)
c22_edopag: char
m132_pcca
c32_anu: char(4)
c32_proyec: char(2)
c32_partid: char(4)
c32_cvecc: char(10)
d32_ptoasg: decimal(14,2)
d32_amplia: decimal(14,2)
d32_reducc: decimal(14,2)
d32_transf: decimal(14,2)
d32_gtocom: decimal(14,2)
d32_gtoeje: decimal(14,2)
m138_movp
c38_anu: char(4)
c38_mes: char(2)
c38_dia: char(2)
c38_proyec: char(2)
c38_partid: char(4)
c38_cveprg: char(4)
s38_folio: smallint
c38_tipmov: char(2)
d38_impmov: decimal(14,2)
d110_parv09
c10_anuafe: char(4)
c10_mesafe: char(2)
c10_numvum: char(8)
s10_secpar: smallint
c10_conpre: char
c10_partid: char(4)
c10_cta: char(5)
c10_scta: char(4)
c10_sscta: char(4)
c10_ssscta: char(4)
c10_tipmov: char
c10_refe: char(10)
d10_imppar: decimal(14,2)
c10_tipdoc: char(2)
c10_edoafe: char
m147_gfrm
c47_anu: char(4)
c47_mes: char(2)
c47_proyec: char(2)
c47_partid: char(4)
c47_tiprov: char
c47_numfon: char(7)
d47_gtoeje: decimal(14,2)
d47_acugej: decimal(14,2)
m154_acub
c54_numbco: char(2)
c54_numcta: char(10)
f54_fecope: date
d54_totdep: decimal(14,2)
d54_totcar: decimal(14,2)
d54_totcre: decimal(14,2)
d54_totche: decimal(14,2)
d54_sdodia: decimal(14,2)
d54_sdoacu: decimal(14,2)
m109_vums
c09_proyec: char(2)
c09_numvum: char(8)
f09_fecreg: date
c09_anuafe: char(4)
c09_mesafe: char(2)
f09_fecpag: date
c09_tipope: char
c09_edovum: char(2)
c09_cvepro: char(7)
c09_tiprov: char
d09_impvum: decimal(14,2)
d09_sdovum: decimal(14,2)
c09_firmae: char(2)
c09_firmar: char(2)
c09_firmaa: char(2)
c09_tidesc: char(2)
c09_despol: char(30)
c09_edoafe: char
c09_origen: char
c09_numche: char(8)
c09_conce1: char(30)
c09_conce2: char(30)
c09_conce3: char(30)
c09_conce4: char(30)
c104_fusa
c04_numvum: char(8)
c04_enuso: char
m150_emis
c50_numsol: char(8)
c50_cvepro: char(7)
c50_tiprov: char
c50_numche: char(8)
c50_nombre: varchar(100)
f50_fereal: date
f50_fecope: date
c50_numbco: char(2)
d50_impche: decimal(14,2)
c50_conce1: char(30)
c50_conce2: char(30)
c50_conce3: char(30)
c50_conce4: char(30)
c50_proyec: char(2)
c50_numvum: char(8)
c50_cvevig: char
c50_concep: char
c50_origen: char
c50_numcta: char(10)
c50_numcxp: char(8)
m106_pgrl
c06_nomcia: char(50)
c06_tipfol: char(3)
c06_darer: char
c06_dindv: char
c06_dprov: char
c06_dfrev: char
c06_dceco: char
c06_dprog: char
c06_txceco: char
c06_txprog: char
c112_feco
f12_fecope: date
c12_cierre: char
f12_fecier: date
c103_faut
c03_anu: char(4)
c03_mes: char(2)
i03_secu: int
d128_prgc26
c28_numcxp: char(8)
s28_secpar: char(2)
c28_partid: char(4)
s28_secprg: smallint
c28_cveprg: char(2)
c28_anuafe: char(4)
c28_mesafe: char(2)
d28_impprg: decimal(14,2)
c28_edoafe: char
d129_ccpp26
c29_numcxp: char(8)
s29_secpar: char(2)
c29_partid: char(4)
s29_secco: smallint
c29_cvecon: char(10)
c29_anuafe: char(4)
c29_mesafe: char(2)
d29_impcc: decimal(14,2)
c29_edoafe: char
m126_cxpp
c26_numcxp: char(8)
f26_fecreg: date
c26_proyec: char(2)
c26_edocxp: char(2)
c26_cvepro: char(7)
c26_tiprov: char
d26_impcxp: decimal(14,2)
d26_sdocxp: decimal(14,2)
c26_descxp: char(30)
c26_edoafe: char
iste
rfc: char(13)
nombre: char(50)
descuento: decimal(12,2)
2, 1 / 2, 2

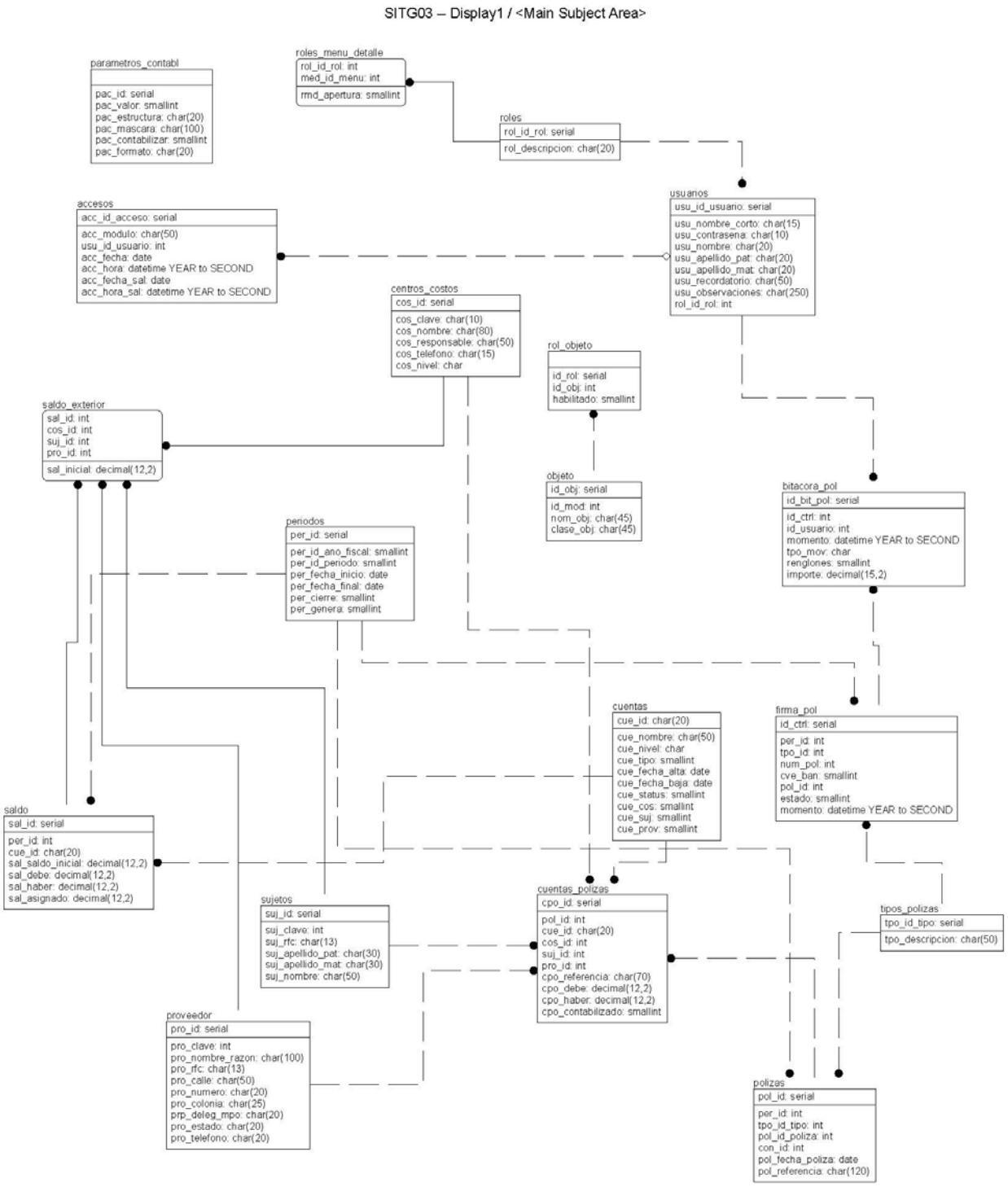
BD1_SITG -- Display1 / <Main Subject Area>



1, 2 / 2, 2


BD1_SITG -- Display1 / <Main Subject Area>
t_paso3
numche: char(8)
numcxp: char(8)
numvum: char(8)
cvevig: char
numsol: char(8)
cvepro: char(7)
nombre: char(30)
fereal: date
numbco: char(2)
numcta: char(10)
impche: decimal(14,2)
conce1: char(30)
conce2: char(30)
conce3: char(30)
conce4: char(30)
origen: char
tabla: char
t_bancos
numbco: char(2)
numcta: char(10)
fecope: date
cvemov: char
refere: char(10)
concep: char(30)
impmov: decimal(14,2)
nombre: char(40)
conce2: char(30)
vumsdesafe
numvum: char(8)
cuantas: int
sf_legis
leg_numero: char(10)
t_bancos2
numbco: char(2)
numcta: char(10)
fecope: date
cvemov: char
refere: char(10)
concep: char(30)
impmov: decimal(14,2)
nombre: char(40)
m109_vums1
c09_proyec: char(2)
c09_numvum: char(8)
f09_fecreg: date
c09_anuafe: char(4)
c09_mesafe: char(2)
f09_fecpag: date
c09_tipope: char
c09_edovum: char(2)
c09_cvepro: char(7)
c09_tiprov: char
d09_impvum: decimal(14,2)
d09_sdovum: decimal(14,2)
c09_firmae: char(2)
c09_firmar: char(2)
c09_firmaa: char(2)
c09_tidesc: char(2)
c09_despol: char(30)
c09_edoafe: char
c09_origen: char
c09_numche: char(8)
c09_pergas: smallint
t_paso
numche: char(8)
numsol: char(8)
numcxp: char(8)
cvepro: char(7)
tiprov: char
nombre: char(30)
fereal: date
fecope: date
numbco: char(2)
numcta: char(10)
impche: decimal(14,2)
numvum: char(8)
conce1: char(30)
conce2: char(30)
conce3: char(30)
conce4: char(30)
cvevig: char
origen: char
tabla: char
concep: char
d116_arev09
c16_anuafe: char(4)
c16_mesafe: char(2)
c16_numvum: char(8)
s16_secpar: smallint
c16_partid: char(4)
s16_secae: smallint
c16_cveae: char(2)
d16_impae: decimal(14,2)
c16_edoafe: char
t_m133_pccm1
c33_anu: char(4)
c33_mes: char(2)
c33_proyec: char(2)
c33_partid: char(4)
c33_cvecc: char(10)
d33_ptoasg: decimal(14,2)
d33_amplia: decimal(14,2)
d33_acuamp: decimal(14,2)
d33_reducc: decimal(14,2)
d33_acured: decimal(14,2)
d33_transf: decimal(14,2)
d33_acutra: decimal(14,2)
d33_gtocom: decimal(14,2)
d33_acugco: decimal(14,2)
d33_gtoeje: decimal(14,2)
d33_acugej: decimal(14,2)
t_paso2
numche: char(8)
numsol: char(8)
numcxp: char(8)
cvepro: char(7)
nombre: char(30)
fereal: date
fecope: date
numbco: char(2)
numcta: char(10)
impche: decimal(14,2)
numvum: char(8)
conce1: char(30)
conce2: char(30)
conce3: char(30)
conce4: char(30)
cvevig: char
origen: char
tabla: char
concep: char
t_paso4
numsol: char(8)
cvepro: char(7)
numche: char(8)
nombre: char(30)
fecope: date
numbco: char(2)
numcta: char(10)
numcxp: char(8)
numvum: char(8)
impche: decimal(14,2)
conce1: char(30)
cvevig: char
m113_part_old
c13_partid: char(4)
c13_despar: char(30)
c13_cta: char(5)
c13_scta: char(4)
c13_sscta: char(4)
c13_ssscta: char(4)
c13_tipctr: char
c13_activa: char
c13_preint: char
sustituto
cencos: char(4)
newcencos: char(7)
descri: char(50)
d127_parc26
c27_numcxp: char(8)
s27_secpar: char(2)
c27_partid: char(4)
c27_anuafe: char(4)
c27_mesafe: char(2)
d27_imppar: decimal(14,2)
c27_tipdoc: char(2)
c27_edoafe: char
2, 2 / 2, 2

Base de datos de contabilidad “sitgNN”.



7 EVALUACIÓN DE PROCESOS

Se analizaron los procesos existentes y se encontraron los siguientes resultados:

Proceso	Programa fuente	Análisis
Cálculo de la Nómina Efectúa el cálculo de la nómina	/sistemas/nomina/P3CALCRP.4g /sistemas/nomina/P3AUTOMA.4gl /sistemas/nomina/P3PASMO.4gl /sistemas/nomina/P3AUTOFI.4gl	Se requieren modificar estos programas para sustituir el proceso del cálculo del ISR, hacer los cálculos del ISSSTE y FOVISSSTE, integrar los préstamos, tiempo extra, faltas, servicio de guardería, y todos los descuentos para pago a terceros.
Empleados Mantenimiento al catálogo de Empleados	/sistemas/nomina/A3MAEEMP.4gl	Se requiere agregar a la base de datos la tabla que contendrá el kardex de los empleados y reestructurar todo el programa para manejar la nueva tabla para el registro del kardex, el seguro de separación individualizado, crédito hipotecario y ahorro solidario.
Movimientos de Percepciones y Deducciones fijos Mantenimiento a los conceptos fijos de los Empleados	/sistemas/nomina/FIJOS.4gl	Muchos de los conceptos que hay que calcular se encuentran como valores fijos que se cargan quincena a quincena, sin embargo, deberán ser calculados y evaluados cada quincena para aplicarlos correctamente, como los préstamos en general, servicio de guardería y pensión alimenticia; por lo cual deberán crearse nuevos procesos para dar mantenimiento a estos conceptos y sólo permitir visualizarlos por este proceso.
Super Bajas Mantenimiento a conceptos de una nómina ya calculada	/sistemas/nomina/SUPERBAJA.4gl	Se requiere desarrollar un nuevo programa que permita modificar todos los conceptos a la vez, presentándose como se ven en el recibo, con sumas de percepciones, deducciones y neto.
Periodos de Nómina Crea los periodos de las nóminas a procesar	/sistemas/nomina/A3PERCTL.4gl	De acuerdo a la evaluación de la codificación de los periodos de nómina y sus respectivas fechas, se propone cambiar la codificación y que el programa genere la codificación correcta con base en las fechas iniciales de las quincenas.
CONSAR Genera archivo para la CONSAR	/sistemas/nomina/consar.4gl	Crear nuevos procesos para generar movimientos de altas, bajas y modificaciones por separado.
ISSSTE Generación archivo ISSSTE	/sistemas/nomina/ENVISSSTE.4gl	Sólo se requiere ajustar el LAYOUT a las nuevas especificaciones.
FOVISSSTE Generación archivo FOVISSSTE	/sistemas/nomina/ENVFOVISTE.4gl	Sólo se requiere ajustar el LAYOUT a las nuevas especificaciones.
Asegurados Generación archivo de Registro de Asegurados	/sistemas/nomina/ASEGURADO1.4gl	Sólo se requiere ajustar el LAYOUT a las nuevas especificaciones.
Distribución de la Nómina Genera reporte a detalle de una nómina	/sistemas/nomina/R3PRENOM.4gl	El reporte tarda mucho en generarse y se detectó que las tablas de detalle de percepciones y deducciones no cuentan con los índices correctos para rápida consulta. Se deberán crear los índices correspondientes.
Resumen de la Nómina por área Genera reporte resumen de una nómina por área	/sistemas/nomina/RESUMENAREA.4gl	Igual que el proceso anterior.
Recibos en papel Stock Genera reporte de recibos de nómina para imprimir en papel Stock	/sistemas/nomina/R3RECPAP.4gl	Debido a que este proceso genera reportes de tipo texto plano, no es posible generar impresiones con diferentes tipos de letras y tamaños y posicionarlos en determinadas áreas de la hoja del recibo, se optará por desarrollar este proceso en Delphi.
Impresión de Cheques Genera la impresión de los cheques que ya fueron seleccionados	/sistemas/finanzas/A1IMPCHE.4gl	Para el caso de la impresión de cheques es factible reutilizar este programa, agregándole la funcionalidad de la impresión de las cuentas contables, en la póliza del cheque.
Reimpresión Emis Reimpresión de queches en emis	/sistemas/finanzas/A1CHEREI.4gl	Igual que el proceso anterior.
Impresión de Cheques de Nómina Permite imprimir lo cheques que se encuentran generados	/sistemas/nomina/A3IMPCHE.4gl	Igual que el proceso anterior.
Reimpresión de Cheques de Nómina Permite reimprimir los cheques que se encuentran generados	/sistemas/nomina/A3CHEREI.4gl	Igual que el proceso anterior.
Impresión Bancomer Cheques Bancomer de 3 por hoja	/sistemas/finanzas/CHEQUE20.4gl	Este proceso desaparece del sistema.

8 INGENIERÍA INVERSA Y REESTRUCTURACIÓN DE DOCUMENTOS

A continuación se muestra el fragmento de código original de un programa y la reestructuración del mismo:

Fragmento de código original del programa `A3PARSI1.4gl`:

```
############################################################
##
## Esta funcion permite realizar cambios a los datos de la tabla
## m320_dsac
##
FUNCTION cambios()
  DEFINE r_undo RECORD LIKE d304_prgr.*, # Sirve para guardar el registro
                                         # antes de que se modifique
         BAND   SMALLINT,
        w_d304_prgr   RECORD LIKE d304_prgr.*
  LET INT_FLAG = FALSE
  INPUT BY NAME r_d304_prgr.c04_tipnom
  AFTER FIELD c04_tipnom
  IF r_d304_prgr.c04_tipnom IS NULL
     THEN
           CALL mensajes(15)
           NEXT FIELD c04_tipnom
  END IF
  IF LENGTH(r_d304_prgr.c04_tipnom) <> 2
     THEN
           CALL mensajes(22)
           NEXT FIELD c04_tipnom
  END IF
  IF valtipnom()
     THEN
          CALL mensajes(20)
  END IF
  SELECT * INTO r_c304_prgr.* FROM c304_prgr
  WHERE c04_tipnom = r_d304_prgr.c04_tipnom
  IF STATUS = NOTFOUND
     THEN
           CALL mensajes(28)
           NEXT FIELD c04_tipnom
  END IF
  DISPLAY BY NAME r_c304_prgr.c04_nombre
END INPUT
IF INT_FLAG
   THEN
  CALL mensajes(3)
  CALL reset()
  RETURN
END IF
  SELECT * INTO r_d304_prgr.* FROM d304_prgr
  WHERE c04_tipnom = r_c304_prgr.c04_tipnom
  IF STATUS = NOTFOUND
     THEN
          LET r_d304_prgr.c04_tipnom = r_c304_prgr.c04_tipnom
          LET r_d304_prgr.s04_redond = 1
          LET r_d304_prgr.d04_depare = 0
          LET r_d304_prgr.d04_despen = 0
          LET r_d304_prgr.d04_prefin = 0
          LET r_d304_prgr.d04_presoc = 0
          LET r_d304_prgr.d04_pagesp = 0
          LET r_d304_prgr.d04_otrper = 0
          LET r_d304_prgr.d04_otrded = 0
          LET r_d304_prgr.d04_subsid = 0
          LET r_d304_prgr.d04_facser = 0
          LET r_d304_prgr.d04_facpen = 0
          LET r_d304_prgr.d04_segvid = 0
          LET r_d304_prgr.d04_segret = 0
          INSERT INTO d304_prgr values(r_d304_prgr.*)
          LET p_status = STATUS
     CASE
          WHEN  p_status = -100 OR p_status = -239
          CALL  mensajes(14)
          WHEN  p_status = 0
     OTHERWISE
          ERROR "ERROR", p_status USING "-<<<<<<" , "A Ocurrido"
     END CASE
  END IF
  LET r_undo.* = r_d304_prgr.*
  LET  w_d304_prgr.*       = r_d304_prgr.*
  INPUT  BY NAME r_d304_prgr.s04_redond,
                 r_d304_prgr.d04_depare,
                 r_d304_prgr.d04_despen,
                 r_d304_prgr.d04_prefin,
                 r_d304_prgr.d04_presoc,
                 r_d304_prgr.d04_pagesp,
                 r_d304_prgr.d04_otrper,
                 r_d304_prgr.d04_otrded,
                 r_d304_prgr.d04_subsid,
                 r_d304_prgr.d04_facser,
                 r_d304_prgr.d04_facpen,
                 r_d304_prgr.d04_segvid,
```

```
              r_d304_prgr.d04_segret
WITHOUT DEFAULTS
 AFTER FIELD s04_redond
  IF r_d304_prgr.s04_redond  IS NULL THEN
    CALL mensajes(15)
    NEXT FIELD s04_redond
  END IF
  CASE
       WHEN r_d304_prgr.s04_redond = 1
            NEXT FIELD d04_depare
       WHEN r_d304_prgr.s04_redond = 10
            NEXT FIELD d04_depare
       WHEN r_d304_prgr.s04_redond = 100
            NEXT FIELD d04_depare
       WHEN r_d304_prgr.s04_redond = 1000
            NEXT FIELD d04_depare
       OTHERWISE
            CALL mensajes(30)
            NEXT FIELD s04_redond
  END CASE
 AFTER FIELD d04_depare
 IF r_d304_prgr.d04_depare IS NULL THEN
    CALL mensajes(15)
    NEXT FIELD  d04_depare
 END IF
 AFTER FIELD d04_despen
 IF r_d304_prgr.d04_despen IS NULL THEN
    CALL mensajes(15)
    NEXT FIELD  d04_despen
 END IF
 IF r_d304_prgr.d04_despen < 0
    THEN
         CALL mensajes(29)
         NEXT FIELD d04_despen
 END IF
 AFTER FIELD d04_prefin
 IF r_d304_prgr.d04_prefin IS NULL THEN
 CALL mensajes(15)
 NEXT FIELD d04_prefin
 END IF
 IF r_d304_prgr.d04_prefin < 0
    THEN
         CALL mensajes(29)
         NEXT FIELD d04_prefin
 END IF
 AFTER FIELD d04_presoc
 IF r_d304_prgr.d04_presoc IS NULL THEN
     CALL mensajes(15)
     NEXT FIELD d04_presoc
 END IF
 IF r_d304_prgr.d04_presoc < 0
    THEN
         CALL mensajes(29)
         NEXT FIELD d04_presoc
 END IF
 AFTER FIELD d04_pagesp
 IF r_d304_prgr.d04_pagesp IS NULL THEN
     CALL mensajes(15)
     NEXT FIELD d04_pagesp
 END IF
 IF r_d304_prgr.d04_pagesp < 0
    THEN
         CALL mensajes(29)
         NEXT FIELD d04_pagesp
 END IF
 AFTER FIELD d04_otrper
 IF r_d304_prgr.d04_otrper IS NULL THEN
     CALL mensajes(15)
     NEXT FIELD d04_otrper
 END IF
 IF r_d304_prgr.d04_otrper < 0
    THEN
         CALL mensajes(29)
         NEXT FIELD d04_otrper
 END IF
 AFTER FIELD d04_otrded
 IF r_d304_prgr.d04_otrded IS NULL THEN
     CALL mensajes(15)
     NEXT FIELD d04_otrded
 END IF
 IF r_d304_prgr.d04_otrded < 0
    THEN
         CALL mensajes(29)
         NEXT FIELD d04_otrded
 END IF
 AFTER FIELD d04_subsid
 IF r_d304_prgr.d04_subsid IS NULL THEN
     CALL mensajes(15)
     NEXT FIELD d04_subsid
 END IF
 IF r_d304_prgr.d04_subsid < 0
    THEN
         CALL mensajes(29)
         NEXT FIELD d04_subsid
 END IF
 AFTER FIELD d04_facser
 IF r_d304_prgr.d04_facser IS NULL THEN
     CALL mensajes(15)
```

```
        NEXT FIELD d04_facser
    END IF
    IF r_d304_prgr.d04_facser < 0
       THEN
            CALL mensajes(29)
            NEXT FIELD d04_facser
    END IF
    AFTER FIELD d04_facpen
    IF r_d304_prgr.d04_facpen IS NULL THEN
        CALL mensajes(15)
        NEXT FIELD d04_facpen
    END IF
    IF r_d304_prgr.d04_facpen < 0
       THEN
            CALL mensajes(29)
            NEXT FIELD d04_facpen
    END IF
    AFTER FIELD d04_segvid
    IF r_d304_prgr.d04_segvid IS NULL THEN
        CALL mensajes(15)
        NEXT FIELD d04_segvid
    END IF
    IF r_d304_prgr.d04_segvid < 0
       THEN
            CALL mensajes(29)
            NEXT FIELD d04_segvid
    END IF
    AFTER FIELD d04_segret
    IF r_d304_prgr.d04_segret IS NULL THEN
        CALL mensajes(15)
        NEXT FIELD d04_segret
    END IF
    IF r_d304_prgr.d04_segret < 0
       THEN
            CALL mensajes(29)
            NEXT FIELD d04_segret
    END IF
END INPUT
IF NOT INT_FLAG THEN
    #WHENEVER ERROR CONTINUE
    UPDATE d304_prgr
      SET d304_prgr.* = r_d304_prgr.*
      WHERE c04_tipnom = r_d304_prgr.c04_tipnom
    CASE
      WHEN STATUS = 0
        CALL mensajes(6)
        CALL reset()
      WHEN STATUS = 100
        CALL mensajes(17)
      OTHERWISE
        ERROR "ERROR: ", STATUS USING "-<<<<", "HA OCURRIDO"
        EXIT PROGRAM
    END CASE
    #WHENEVER ERROR CALL llama_err
ELSE
     LET INT_FLAG = FALSE
     LET r_d304_prgr.* = r_undo.*
  DISPLAY BY NAME r_d304_prgr.*
  CALL mensajes(3)
  CALL reset()
END IF
END FUNCTION
############################################################
```

Fragmento de código reestructurado del programa `A3PARSI1.4gl`:

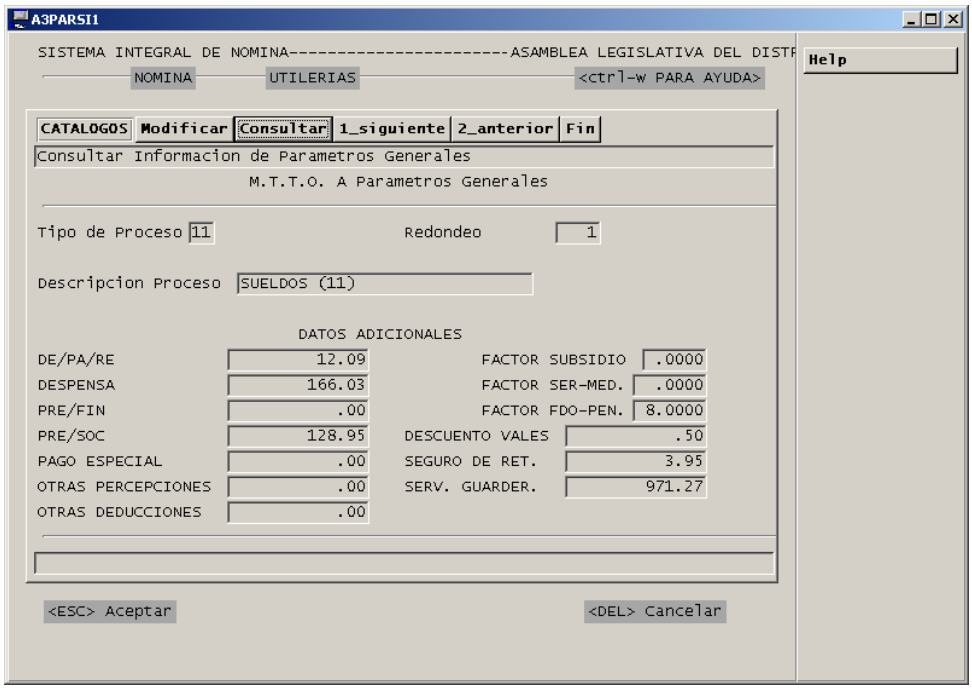
```
################################################################################
## Esta funcion permite realizar cambios a los datos de la tabla m320_dsac
################################################################################
FUNCTION cambios()
   DEFINE
      r_undo            RECORD LIKE d304_prgr.*, # Sirve para guardar el registro
                                                 # antes de que se modifique
      BAND              SMALLINT,
      w_d304_prgr       RECORD LIKE d304_prgr.*


  LET INT_FLAG = FALSE

  INPUT BY NAME r_d304_prgr.c04_tipnom

      AFTER FIELD c04_tipnom
         IF r_d304_prgr.c04_tipnom IS NULL THEN
            CALL mensajes(15)
            NEXT FIELD c04_tipnom
         END IF
         IF LENGTH(r_d304_prgr.c04_tipnom) <> 2 THEN
            CALL mensajes(22)
            NEXT FIELD c04_tipnom
         END IF
         IF valtipnom() THEN
            CALL mensajes(20)
         END IF

         SELECT *
            INTO r_c304_prgr.*
```

```
            FROM c304_prgr
            WHERE c04_tipnom = r_d304_prgr.c04_tipnom

         IF STATUS = NOTFOUND THEN
            CALL mensajes(28)
            NEXT FIELD c04_tipnom
         END IF

         DISPLAY BY NAME r_c304_prgr.c04_nombre

   END INPUT

   IF INT_FLAG THEN
      CALL mensajes(3)
      CALL reset()
      RETURN
   END IF

   SELECT *
      INTO r_d304_prgr.*
      FROM d304_prgr
      WHERE c04_tipnom = r_c304_prgr.c04_tipnom

   IF STATUS = NOTFOUND THEN
      LET r_d304_prgr.c04_tipnom = r_c304_prgr.c04_tipnom
      LET r_d304_prgr.s04_redond = 1
      LET r_d304_prgr.d04_depare = 0
      LET r_d304_prgr.d04_despen = 0
      LET r_d304_prgr.d04_prefin = 0
      LET r_d304_prgr.d04_presoc = 0
      LET r_d304_prgr.d04_pagesp = 0
      LET r_d304_prgr.d04_otrper = 0
      LET r_d304_prgr.d04_otrded = 0
      LET r_d304_prgr.d04_subsid = 0
      LET r_d304_prgr.d04_facser = 0
      LET r_d304_prgr.d04_facpen = 0
      LET r_d304_prgr.d04_segvid = 0
      LET r_d304_prgr.d04_segret = 0

      INSERT INTO d304_prgr values(r_d304_prgr.*)

      LET p_status = STATUS

      CASE
         WHEN p_status = -100 OR p_status = -239
            CALL  mensajes(14)
         WHEN p_status != 0
            ERROR "ERROR", p_status USING "-<<<<<<" , "A Ocurrido"
      END CASE
   END IF

   LET r_undo.* = r_d304_prgr.*

   LET w_d304_prgr.* = r_d304_prgr.*

   INPUT BY NAME r_d304_prgr.s04_redond,
                 r_d304_prgr.d04_depare,
                 r_d304_prgr.d04_despen,
                 r_d304_prgr.d04_prefin,
                 r_d304_prgr.d04_presoc,
                 r_d304_prgr.d04_pagesp,
                 r_d304_prgr.d04_otrper,
                 r_d304_prgr.d04_otrded,
                 r_d304_prgr.d04_subsid,
                 r_d304_prgr.d04_facser,
                 r_d304_prgr.d04_facpen,
                 r_d304_prgr.d04_segvid,
                 r_d304_prgr.d04_segret,
                 r_d304_prgr.d04_guarderia
      WITHOUT DEFAULTS

      AFTER FIELD s04_redond
         IF r_d304_prgr.s04_redond IS NULL THEN
            CALL mensajes(15)
            NEXT FIELD s04_redond
         END IF

         CASE
            WHEN r_d304_prgr.s04_redond = 1
               NEXT FIELD d04_depare
            WHEN r_d304_prgr.s04_redond = 10
               NEXT FIELD d04_depare
            WHEN r_d304_prgr.s04_redond = 100
               NEXT FIELD d04_depare
            WHEN r_d304_prgr.s04_redond = 1000
               NEXT FIELD d04_depare
            OTHERWISE
               CALL mensajes(30)
               NEXT FIELD s04_redond
         END CASE

      AFTER FIELD d04_depare
         IF r_d304_prgr.d04_depare IS NULL THEN
            CALL mensajes(15)
            NEXT FIELD d04_depare
         END IF
         IF r_d304_prgr.d04_depare < 0 THEN
            CALL mensajes(29)
```

```
      NEXT FIELD d04_depare
   END IF

AFTER FIELD d04_despen
   IF r_d304_prgr.d04_despen IS NULL THEN
      CALL mensajes(15)
      NEXT FIELD  d04_despen
   END IF
   IF r_d304_prgr.d04_despen < 0 THEN
      CALL mensajes(29)
      NEXT FIELD d04_despen
   END IF

AFTER FIELD d04_prefin
   IF r_d304_prgr.d04_prefin IS NULL THEN
      CALL mensajes(15)
      NEXT FIELD d04_prefin
   END IF
   IF r_d304_prgr.d04_prefin < 0 THEN
      CALL mensajes(29)
      NEXT FIELD d04_prefin
   END IF

AFTER FIELD d04_presoc
   IF r_d304_prgr.d04_presoc IS NULL THEN
      CALL mensajes(15)
      NEXT FIELD d04_presoc
   END IF
   IF r_d304_prgr.d04_presoc < 0 THEN
      CALL mensajes(29)
      NEXT FIELD d04_presoc
   END IF

AFTER FIELD d04_pagesp
   IF r_d304_prgr.d04_pagesp IS NULL THEN
      CALL mensajes(15)
      NEXT FIELD d04_pagesp
   END IF
   IF r_d304_prgr.d04_pagesp < 0 THEN
      CALL mensajes(29)
      NEXT FIELD d04_pagesp
   END IF

AFTER FIELD d04_otrper
   IF r_d304_prgr.d04_otrper IS NULL THEN
      CALL mensajes(15)
      NEXT FIELD d04_otrper
   END IF
   IF r_d304_prgr.d04_otrper < 0 THEN
      CALL mensajes(29)
      NEXT FIELD d04_otrper
   END IF

AFTER FIELD d04_otrded
   IF r_d304_prgr.d04_otrded IS NULL THEN
      CALL mensajes(15)
      NEXT FIELD d04_otrded
   END IF
   IF r_d304_prgr.d04_otrded < 0 THEN
      CALL mensajes(29)
      NEXT FIELD d04_otrded
   END IF

AFTER FIELD d04_subsid
   IF r_d304_prgr.d04_subsid IS NULL THEN
      CALL mensajes(15)
      NEXT FIELD d04_subsid
   END IF
   IF r_d304_prgr.d04_subsid < 0 THEN
      CALL mensajes(29)
      NEXT FIELD d04_subsid
   END IF

AFTER FIELD d04_facser
   IF r_d304_prgr.d04_facser IS NULL THEN
      CALL mensajes(15)
      NEXT FIELD d04_facser
   END IF
   IF r_d304_prgr.d04_facser < 0 THEN
      CALL mensajes(29)
      NEXT FIELD d04_facser
   END IF

AFTER FIELD d04_facpen
   IF r_d304_prgr.d04_facpen IS NULL THEN
      CALL mensajes(15)
      NEXT FIELD d04_facpen
   END IF
   IF r_d304_prgr.d04_facpen < 0 THEN
      CALL mensajes(29)
      NEXT FIELD d04_facpen
   END IF

AFTER FIELD d04_segvid
   IF r_d304_prgr.d04_segvid IS NULL THEN
      CALL mensajes(15)
      NEXT FIELD d04_segvid
   END IF
```

```
         IF r_d304_prgr.d04_segvid < 0 THEN
            CALL mensajes(29)
            NEXT FIELD d04_segvid
         END IF

      AFTER FIELD d04_segret
         IF r_d304_prgr.d04_segret IS NULL THEN
            CALL mensajes(15)
            NEXT FIELD d04_segret
         END IF
         IF r_d304_prgr.d04_segret < 0 THEN
            CALL mensajes(29)
            NEXT FIELD d04_segret
         END IF

      AFTER FIELD d04_guarderia
         IF r_d304_prgr.d04_guarderia IS NULL THEN
            CALL mensajes(15)
            NEXT FIELD d04_guarderia
         END IF
         IF r_d304_prgr.d04_guarderia < 0 THEN
            CALL mensajes(29)
            NEXT FIELD d04_guarderia
         END IF

   END INPUT

   IF NOT INT_FLAG THEN
      UPDATE d304_prgr
         SET d304_prgr.* = r_d304_prgr.*
         WHERE c04_tipnom = r_d304_prgr.c04_tipnom
      CASE
         WHEN STATUS = 0
            CALL mensajes(6)
            CALL reset()
         WHEN STATUS = 100
            CALL mensajes(17)
         OTHERWISE
            ERROR "ERROR: ", STATUS USING "-<<<<", "HA OCURRIDO"
            EXIT PROGRAM
      END CASE
   ELSE
      LET INT_FLAG = FALSE
      LET r_d304_prgr.* = r_undo.*
      DISPLAY BY NAME r_d304_prgr.*
      CALL mensajes(3)
      CALL reset()
   END IF

END FUNCTION
###############################################################################
```

En este caso en particular se encontró que este programa utiliza la forma “F3PARSI1.per” como interfaz, para darle mantenimiento a la tabla “d304_prgr” de la base de datos “bd3_nomi”; así mismo valida que los datos introducidos no sean nulos ni negativos.



Se aplica el mismo procedimiento a todos los programas que necesitan ser sometidos a la reingeniería de software.

9 REESTRUCTURACIÓN DE CÓDIGO Y DATOS

A continuación se muestra el fragmento de código original de un programa y la reestructuración del mismo (**Reingeniería**):

Fragmento de código original del programa `A1IMPCHE.4gl`:

```
FOREACH c_ptr INTO r_m150_emis.*           ##  IMPRESION Y ACTUALIZACION
IF r_m150_emis.c50_origen = 5 THEN
   SELECT d26_sdocxp INTO p_sdocxp
   FROM m126_cxpp
   WHERE c26_numcxp = r_m150_emis.c50_numcxp
   IF r_m150_emis.d50_impche > p_sdocxp THEN
      CONTINUE FOREACH
   END IF
END IF

 LET p_primeravez = "1"

 IF r_m150_emis.c50_origen = "1" OR r_m150_emis.c50_origen = "3" THEN
   IF r_m150_emis.c50_concep = "3" THEN
      IF validacta(r_m150_emis.c50_tiprov) THEN
         CALL mensajes(26)
         CONTINUE FOREACH
      END IF
   ELSE
      IF validacta(r_m150_emis.c50_tiprov) THEN
        CALL mensajes(26)
        CONTINUE FOREACH
      END IF
   END IF
END IF
            ################################################
            #  CUENTAS POR PAGAR (ASAMBLEA):               #
            #  Se genera una poliza de: GASTOS vs BANCOS  #
            ################################################

IF r_m150_emis.c50_origen = "2" OR r_m150_emis.c50_origen > "3"  THEN
   IF validacta(r_m150_emis.c50_tiprov) THEN
     CALL mensajes(26)
     CONTINUE FOREACH
   END IF
   IF r_m150_emis.c50_origen = "5" THEN
      IF validacxp() THEN
        CALL mensajes(27)
        CONTINUE FOREACH
      END IF
   END IF
END IF

   LET p_sierror = "0"
   LET p_huboerror = 0
   LET w_i50_numche = w_i50_numche + 1
   LET r_m150_emis.c50_numche = w_i50_numche
   CASE
     WHEN LENGTH (r_m150_emis.c50_numche) = 1
       LET r_m150_emis.c50_numche = "0000000",r_m150_emis.c50_numche CLIPPED
     WHEN LENGTH (r_m150_emis.c50_numche) = 2
       LET r_m150_emis.c50_numche = "000000",r_m150_emis.c50_numche CLIPPED
     WHEN LENGTH (r_m150_emis.c50_numche) = 3
       LET r_m150_emis.c50_numche = "00000",r_m150_emis.c50_numche CLIPPED
     WHEN LENGTH (r_m150_emis.c50_numche) = 4
       LET r_m150_emis.c50_numche = "0000",r_m150_emis.c50_numche CLIPPED
     WHEN LENGTH (r_m150_emis.c50_numche) = 5
       LET r_m150_emis.c50_numche = "000",r_m150_emis.c50_numche CLIPPED
     WHEN LENGTH (r_m150_emis.c50_numche) = 6
       LET r_m150_emis.c50_numche = "00",r_m150_emis.c50_numche CLIPPED
     WHEN LENGTH (r_m150_emis.c50_numche) = 7
       LET r_m150_emis.c50_numche = "0",r_m150_emis.c50_numche CLIPPED
     WHEN LENGTH (r_m150_emis.c50_numche) = 8
       LET r_m150_emis.c50_numche = r_m150_emis.c50_numche CLIPPED
   END CASE
   LET p_sierror = "0"
   BEGIN WORK                          # INICIA TRANSACCION DE CONTABILIDAD
     CALL contabiliza()
     IF p_sierror = "1" THEN
        CONTINUE FOREACH
     END IF
```

Fragmento de código reestructurado del programa `A1IMPCHE.4gl`:

Se indentó el código y se modificó la estructura de datos para optimizar su ejecución

```
-- IMPRESION Y ACTUALIZACION

FOREACH c_ptr
   INTO r_m150_emis.*

   SELECT *
      INTO r_m118_bene.*
      FROM m118_bene
      WHERE c18_cvepro = r_m150_emis.c50_cvepro AND
            c18_tiprov = r_m150_emis.c50_tiprov

   IF r_m150_emis.c50_origen = 5 THEN
      SELECT d26_sdocxp
         INTO p_sdocxp
         FROM m126_cxpp
         WHERE c26_numcxp = r_m150_emis.c50_numcxp

      IF r_m150_emis.d50_impche > p_sdocxp THEN
         CONTINUE FOREACH
      END IF
   END IF

   IF r_m150_emis.c50_origen = "1" OR r_m150_emis.c50_origen = "3" THEN
      IF r_m150_emis.c50_concep = "3" THEN
         IF validacta(r_m150_emis.c50_tiprov) THEN
            CALL mensajes(26)
            CONTINUE FOREACH
         END IF
      ELSE
         IF validacta(r_m150_emis.c50_tiprov) THEN
            CALL mensajes(26)
            CONTINUE FOREACH
         END IF
      END IF
   END IF


   -- CUENTAS POR PAGAR (ASAMBLEA): Se genera una poliza de: GASTOS vs BANCOS

   IF r_m150_emis.c50_origen = "2" OR r_m150_emis.c50_origen > "3"  THEN
      IF validacta(r_m150_emis.c50_tiprov) THEN
        CALL mensajes(26)
        CONTINUE FOREACH
      END IF
      IF r_m150_emis.c50_origen = "5" THEN
         IF validacxp() THEN
           CALL mensajes(27)
           CONTINUE FOREACH
         END IF
      END IF
   END IF

   LET p_sierror = "0"
   LET p_huboerror = 0
   LET w_i50_numche = w_i50_numche + 1
   LET r_m150_emis.c50_numche = w_i50_numche USING '&&&&&&&&'

   LET p_sierror = "0"

   BEGIN WORK                          # INICIA TRANSACCION DE CONTABILIDAD

   CALL contabiliza()
   IF p_sierror = "1" THEN
      ROLLBACK WORK
      CONTINUE FOREACH
   END IF
```

En donde el código original:

```
LET r_m150_emis.c50_numche = w_i50_numche
CASE
  WHEN LENGTH (r_m150_emis.c50_numche) = 1
    LET r_m150_emis.c50_numche = "0000000",r_m150_emis.c50_numche CLIPPED
  WHEN LENGTH (r_m150_emis.c50_numche) = 2
    LET r_m150_emis.c50_numche = "000000",r_m150_emis.c50_numche CLIPPED
  WHEN LENGTH (r_m150_emis.c50_numche) = 3
    LET r_m150_emis.c50_numche = "00000",r_m150_emis.c50_numche CLIPPED
  WHEN LENGTH (r_m150_emis.c50_numche) = 4
    LET r_m150_emis.c50_numche = "0000",r_m150_emis.c50_numche CLIPPED
  WHEN LENGTH (r_m150_emis.c50_numche) = 5
    LET r_m150_emis.c50_numche = "000",r_m150_emis.c50_numche CLIPPED
  WHEN LENGTH (r_m150_emis.c50_numche) = 6
    LET r_m150_emis.c50_numche = "00",r_m150_emis.c50_numche CLIPPED
  WHEN LENGTH (r_m150_emis.c50_numche) = 7
    LET r_m150_emis.c50_numche = "0",r_m150_emis.c50_numche CLIPPED
  WHEN LENGTH (r_m150_emis.c50_numche) = 8
    LET r_m150_emis.c50_numche = r_m150_emis.c50_numche CLIPPED
END CASE
```

Se reemplazó por el código (**Refabricación**, **Refinamiento y simplificación**):

```
LET r_m150_emis.c50_numche = w_i50_numche USING '&&&&&&&&'
```

El cual formatea el número contenido en la variable `w_i50_numche` a ocho posiciones, rellenando los espacios a la izquierda con ceros y asignándola al elemento `c50_numche` de la estructura de registro `r_m150_emis (r_m150_emis.c50_numche)`.

Se aplica el mismo procedimiento a todos los programas que necesitan ser sometidos a la reingeniería de software.

10 INGENIERÍA DEL DISEÑO

A continuación se presentan las abstracciones, arquitecturas, patrones, modularidad, ocultación de la información, independencia funcional, refinamiento y refabricación de tres procesos requeridos del sistema: **Impresión del importe en palabras en los cheques y recibos**, **Cálculo del tiempo extra** y **Cálculo del impuesto**.

- **IMPRESIÓN DEL IMPORTE EN PALABRAS EN LOS CHEQUES Y RECIBOS:**

  Este procedimiento incluye **Independencia funcional** y **Refabricación** de la función para convertir un valor numérico (importe) en palabras para la impresión en los cheques y recibos:

  La función `forma_descripcion` utiliza variables globales (además de locales) para el importe a convertir, así como la variable en donde se guarda el texto del importe. El código tiene un “bug” en la generación del texto del importe en las centenas al dejarlas en dos palabras separadas.

```
--------------------------------------------------------------------------
FUNCTION forma_descripcion()

 DEFINE paso  char(15),
        feria char(2),
        valor char(12),
        longi,l,totlo,posic,wpaso,checa,termi SMALLINT,      # trabajo
        wwaux                                 char(20),
        wchar,wchar1                          char(1)

   LET paso  = r_m150_emis.d50_impche
   LET l     = LENGTH(PASO)
   LET valor = paso[1,l-3]
   LET feria = paso[l-1,l]
   LET w_c50_letra = "("
   LET longi = 1
   LET totlo = LENGTH(valor)
   LET posic = 12 - totlo + 1

   IF valor > 0 THEN
      LET checa = TRUE
   ELSE
      LET checa = FALSE
   END IF

   IF totlo = 0 THEN
      LET checa = FALSE
   END IF

   WHILE checa

      LET termi = TRUE
      LET wchar = valor[longi,longi]
      IF posic = 1 OR posic = 4 OR posic = 7 OR posic = 10 THEN
    LET wwaux = ' CIENTO'
         IF wchar > '1' THEN
            LET wwaux = wwaux CLIPPED,'S'
    END IF

    CASE
       when wchar = '0'
          IF valor[longi+1,longi+2] = '00' THEN
        LET longi = longi + 2
```

```
      LET posic = posic + 2
      LET termi = FALSE
        END IF
     when wchar = '1'
        IF valor[longi+1,longi+2] = '00' THEN
      LET posic = posic + 2
      LET longi = longi + 2
      LET wwaux = 'CIEN'
        END IF
             LET l = LENGTH(w_c50_letra)
        LET w_c50_letra = w_c50_letra[1,l],wwaux
     when wchar = '2'
             LET l = LENGTH(w_c50_letra)
        LET w_c50_letra = w_c50_letra[1,l],' DOS',wwaux
     when wchar = '3'
             LET l = LENGTH(w_c50_letra)
        LET w_c50_letra = w_c50_letra[1,l],' TRES',wwaux
     when wchar = '4'
             LET l = LENGTH(w_c50_letra)
        LET w_c50_letra = w_c50_letra[1,l],' CUATRO',wwaux
     when wchar = '5'
             LET l = LENGTH(w_c50_letra)
        LET w_c50_letra = w_c50_letra[1,l],' QUINIENTOS'
     when wchar = '6'
             LET l = LENGTH(w_c50_letra)
        LET w_c50_letra = w_c50_letra[1,l],' SEIS',wwaux
     when wchar = '7'
             LET l = LENGTH(w_c50_letra)
        LET w_c50_letra = w_c50_letra[1,l],' SETE',wwaux
     when wchar = '8'
             LET l = LENGTH(w_c50_letra)
        LET w_c50_letra = w_c50_letra[1,l],' OCHO',wwaux
     when wchar = '9'
             LET l = LENGTH(w_c50_letra)
        LET w_c50_letra = w_c50_letra[1,l],' NOVE',wwaux
END CASE
LET longi = longi + 1
LET posic = posic + 1
  ELSE
IF posic = 2 OR posic = 5 OR posic = 8 OR posic = 11 THEN
   CASE
      when wchar = '1'
    LET wchar1 = valor[longi+1,longi+1]
    CASE
       when wchar1 = '0'
                   LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],' DIEZ'
       when wchar1 = '1'
                   LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],' ONCE'
                when wchar1 = '2'
                   LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],' DOCE'
       when wchar1 = '3'
                   LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],' TRECE'
       when wchar1 = '4'
                   LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],' CATORCE'
       when wchar1 = '5'
                   LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],' QUINCE'
       when wchar1 = '6'
                   LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],' DIECISEIS'
       when wchar1 = '7'
                   LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],' DIECISIETE'
       when wchar1 = '8'
                   LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],' DIECIOCHO'
       when wchar1 = '9'
                   LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],' DIECINUEVE'
    END CASE
    LET longi = longi + 1
    LET posic = posic + 1
      when wchar = '2'
    IF valor[longi+1,longi+1] = '0' THEN
                LET l = LENGTH(w_c50_letra)
       LET w_c50_letra = w_c50_letra[1,l],' VEINTE'
    ELSE
                LET l = LENGTH(w_c50_letra)
       LET w_c50_letra = w_c50_letra[1,l],' VEINTI'
    END IF
      when wchar = '3'
               LET l = LENGTH(w_c50_letra)
    LET w_c50_letra = w_c50_letra[1,l],' TREINTA'
      when wchar = '4'
               LET l = LENGTH(w_c50_letra)
    LET w_c50_letra = w_c50_letra[1,l],' CUARENTA'
      when wchar = '5'
               LET l = LENGTH(w_c50_letra)
    LET w_c50_letra = w_c50_letra[1,l],' CINCUENTA'
      when wchar = '6'
               LET l = LENGTH(w_c50_letra)
    LET w_c50_letra = w_c50_letra[1,l],' SESENTA'
      when wchar = '7'
```

```
                LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],' SETENTA'
       when wchar = '8'
                LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],' OCHENTA'
       when wchar = '9'
                LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],' NOVENTA'
    END CASE
    IF wchar <> '1' THEN
       LET longi = longi + 1
       LET posic = posic + 1
       IF wchar <> '0' AND wchar <> '2' THEN
     LET wwaux = ' Y '
       ELSE
     IF wchar = '2' THEN
        LET wwaux = ''
     ELSE
        LET wwaux = ' '
     END IF
       END IF
       LET wchar = valor[longi,longi]
       CASE
     when wchar = '1'
                LET l = LENGTH(w_c50_letra)
        LET w_c50_letra = w_c50_letra[1,l],wwaux CLIPPED,' UN'
     when wchar = '2'
                LET l = LENGTH(w_c50_letra)
        LET w_c50_letra = w_c50_letra[1,l],wwaux CLIPPED,' DOS'
     when wchar = '3'
                LET l = LENGTH(w_c50_letra)
        LET w_c50_letra = w_c50_letra[1,l],wwaux CLIPPED,' TRES'
     when wchar = '4'
                LET l = LENGTH(w_c50_letra)
      LET w_c50_letra = w_c50_letra[1,l],wwaux CLIPPED,' CUATRO'
     when wchar = '5'
                LET l = LENGTH(w_c50_letra)
       LET w_c50_letra = w_c50_letra[1,l],wwaux CLIPPED,' CINCO'
     when wchar = "6"
                LET l = LENGTH(w_c50_letra)
        LET w_c50_letra = w_c50_letra[1,l],wwaux CLIPPED,' SEIS'
     when wchar = '7'
                LET l = LENGTH(w_c50_letra)
       LET w_c50_letra = w_c50_letra[1,l],wwaux CLIPPED,' SIETE'
     when wchar = '8'
                LET l = LENGTH(w_c50_letra)
        LET w_c50_letra = w_c50_letra[1,l],wwaux CLIPPED,' OCHO'
     when wchar = '9'
                LET l = LENGTH(w_c50_letra)
       LET w_c50_letra = w_c50_letra[1,l],wwaux CLIPPED,' NUEVE'
       END CASE
    END IF
ELSE
    CASE
       when wchar = '1'
                LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],'UN'
       when wchar = '2'
                LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],'DOS'
       when wchar = '3'
                LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],'TRES'
       when wchar = '4'
                LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],'CUATRO'
       when wchar = '5'
                LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],'CINCO'
       when wchar = '6'
                LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],'SEIS'
       when wchar = '7'
                LET l = LENGTH(w_c50_letra)
    LET w_c50_letra = w_c50_letra[1,l],'SIETE'
       when wchar = '8'
                LET l = LENGTH(w_c50_letra)
     LET w_c50_letra = w_c50_letra[1,l],'OCHO'
       when wchar = '9'
                LET l = LENGTH(w_c50_letra)
    LET  w_c50_letra = w_c50_letra[1,l],'NUEVE'
    END CASE
END IF
LET longi = longi + 1
LET posic = posic + 1
  END IF

  IF (posic = 4 OR posic = 10) AND termi THEN
     LET l = LENGTH(w_c50_letra)
LET w_c50_letra = w_c50_letra[1,l],' MIL '
  ELSE
IF posic = 7 THEN
  IF totlo = 7 AND valor[1,1] = '1' THEN
           LET l = LENGTH(w_c50_letra)
       LET w_c50_letra = w_c50_letra[1,l],' MILLON '
   ELSE
           LET l = LENGTH(w_c50_letra)
       LET w_c50_letra = w_c50_letra[1,l],' MILLONES '
```

```
        END IF
     END IF
       END IF

       IF longi > totlo THEN
     LET checa = FALSE
       END IF
    END WHILE

    IF valor = "0" THEN
       LET w_c50_letra = 'CERO'
    END IF
    LET wpaso = LENGTH(w_c50_letra)
IF wpaso > 8 THEN
    IF w_c50_letra[wpaso-8,wpaso] = 'MILLONES ' OR
       w_c50_letra[wpaso-6,wpaso] = 'MILLON ' THEN
       LET wwaux = ' DE PESOS '
    ELSE
       IF valor = 1 THEN
     LET wwaux = ' PESO '
       ELSE
     LET wwaux = ' PESOS '
       END IF
    END IF
ELSE
     IF valor = 1 THEN
       LET wwaux = ' PESO '
     ELSE
       LET wwaux = ' PESOS '
     END IF
END IF

  LET l = LENGTH(w_c50_letra)
  LET w_c50_letra = w_c50_letra[1,l],' MILLONES '
  LET w_c50_letra=w_c50_letra[1,l]," ",wwaux CLIPPED," ",feria,'/100 M.N.)'
END FUNCTION
```

Se reemplazó todo el código (**Refabricación**) y se modificó la estructura de datos para optimizar su ejecución con menos código utilizando un parámetro para el importe y solo con variables locales (**Independencia funcional**), devolviendo el resultado de éste en la función.

```
################################################################################
FUNCTION num_letras(nNumero)
   DEFINE
      nNumero              DECIMAL(11,2),
      cUdc                 CHAR(3),
      cCar                 CHAR(1),
      cNumero              CHAR(12),
      cLetras              CHAR(90),
      i,
      j,
      nVal                 SMALLINT


   LET cNumero = nNumero USING '&&&&&&&&&.&&'
   LET cLetras = ''

   FOR j = 1 TO 3

      LET cUdc = cNumero[1+(j-1)*3, 3+(j-1)*3]


      FOR i = 1 TO 3

         LET cCar = cUdc[i,i]

         CASE
            WHEN i = 1
               LET nVal = cUdc[2,3]
               CASE
                  WHEN cCar = '9'
                     LET cLetras = cLetras CLIPPED, ' NOVECIENTOS'
                  WHEN cCar = '8'
                     LET cLetras = cLetras CLIPPED, ' OCHOCIENTOS'
                  WHEN cCar = '7'
                     LET cLetras = cLetras CLIPPED, ' SETECIENTOS'
                  WHEN cCar = '6'
                     LET cLetras = cLetras CLIPPED, ' SEISCIENTOS'
                  WHEN cCar = '5'
                     LET cLetras = cLetras CLIPPED, ' QUINIENTOS'
                  WHEN cCar = '4'
                     LET cLetras = cLetras CLIPPED, ' CUATROCIENTOS'
                  WHEN cCar = '3'
                     LET cLetras = cLetras CLIPPED, ' TRESCIENTOS'
                  WHEN cCar = '2'
                     LET cLetras = cLetras CLIPPED, ' DOSCIENTOS'
                  WHEN cCar = '1'
                     IF nVal > 0 THEN
                        LET cLetras = cLetras CLIPPED, ' CIENTO'
                     ELSE
                        LET cLetras = cLetras CLIPPED, ' CIEN'
```

```
            END IF
      END CASE

      IF (nVal > 0) AND (LENGTH(cLetras) != 0) AND (cCar != '0') THEN
         LET cLetras = cLetras CLIPPED, ' '
      END IF

   WHEN i = 2
      LET nVal = cUdc[3,3]
      CASE
         WHEN cCar = '9'
            LET cLetras = cLetras CLIPPED, ' NOVENTA'
         WHEN cCar = '8'
            LET cLetras = cLetras CLIPPED, ' OCHENTA'
         WHEN cCar = '7'
            LET cLetras = cLetras CLIPPED, ' SETENTA'
         WHEN cCar = '6'
            LET cLetras = cLetras CLIPPED, ' SESENTA'
         WHEN cCar = '5'
            LET cLetras = cLetras CLIPPED, ' CINCUENTA'
         WHEN cCar = '4'
            LET cLetras = cLetras CLIPPED, ' CUARENTA'
         WHEN cCar = '3'
            LET cLetras = cLetras CLIPPED, ' TREINTA'
         WHEN cCar = '2'
            IF nVal > 0 THEN
               LET cLetras = cLetras CLIPPED, ' VEINTI'
            ELSE
               LET cLetras = cLetras CLIPPED, ' VEINTE'
            END IF
         WHEN cCar = '1'
            IF nVal = 0 THEN
               LET cLetras = cLetras CLIPPED, ' DIEZ'
            ELSE
               IF nVal > 5 THEN
                  LET cLetras = cLetras CLIPPED, ' DIECI'
               ELSE
                  CASE
                     WHEN nVal = 1
                        LET cLetras = cLetras CLIPPED, ' ONCE'
                     WHEN nVal = 2
                        LET cLetras = cLetras CLIPPED, ' DOCE'
                     WHEN nVal = 3
                        LET cLetras = cLetras CLIPPED, ' TRECE'
                     WHEN nVal = 4
                        LET cLetras = cLetras CLIPPED, ' CATORCE'
                     WHEN nVal = 5
                        LET cLetras = cLetras CLIPPED, ' QUINCE'
                  END CASE
               END IF
            END IF
      END CASE

      IF (nVal > 0) and (cCar > '2') THEN
         LET cLetras = cLetras CLIPPED, ' Y '
      END IF

   WHEN i = 3
      LET nVal = cUdc[2,2]
      CASE
         WHEN (cCar = '9')
            IF nVal = 1 OR nVal = 2 THEN
               LET cLetras = cLetras CLIPPED, 'NUEVE'
            ELSE
               LET cLetras = cLetras CLIPPED, ' NUEVE'
            END IF
         WHEN (cCar = '8')
            IF nVal = 1 OR nVal = 2 THEN
               LET cLetras = cLetras CLIPPED, 'OCHO'
            ELSE
               LET cLetras = cLetras CLIPPED, ' OCHO'
            END IF
         WHEN (cCar = '7')
            IF nVal = 1 OR nVal = 2 THEN
               LET cLetras = cLetras CLIPPED, 'SIETE'
            ELSE
               LET cLetras = cLetras CLIPPED, ' SIETE'
            END IF
         WHEN (cCar = '6')
            IF nVal = 1 OR nVal = 2 THEN
               LET cLetras = cLetras CLIPPED, 'SEIS'
            ELSE
               LET cLetras = cLetras CLIPPED, ' SEIS'
            END IF
         WHEN (cCar = '5') AND (nVal != 1)
            IF nVal = 2 THEN
               LET cLetras = cLetras CLIPPED, 'CINCO'
            ELSE
               LET cLetras = cLetras CLIPPED, ' CINCO'
            END IF
         WHEN (cCar = '4') AND (nVal != 1)
            IF nVal = 2 THEN
               LET cLetras = cLetras CLIPPED, 'CUATRO'
            ELSE
               LET cLetras = cLetras CLIPPED, ' CUATRO'
            END IF
         WHEN (cCar = '3') AND (nVal != 1)
            IF nVal = 2 THEN
```

```
                        LET cLetras = cLetras CLIPPED, 'TRES'
                     ELSE
                        LET cLetras = cLetras CLIPPED, ' TRES'
                     END IF
                  WHEN (cCar = '2') AND (nVal != 1)
                     IF nVal = 2 THEN
                        LET cLetras = cLetras CLIPPED, 'DOS'
                     ELSE
                        LET cLetras = cLetras CLIPPED, ' DOS'
                     END IF
                  WHEN (cCar = '1') AND (nVal != 1)
                     IF nVal = 2 THEN
                        LET cLetras = cLetras CLIPPED, 'UN'
                     ELSE
                        LET cLetras = cLetras CLIPPED, ' UN'
                     END IF
               END CASE
         END CASE

      END FOR

      IF cUdc != '000' THEN
         LET nVal = cUdc
         IF j = 2 THEN
            IF nVal = 1 THEN
               LET cLetras = cLetras CLIPPED, 'MIL '
            ELSE
               LET cLetras = cLetras CLIPPED, ' MIL '
            END IF
         ELSE
            IF j = 1 THEN
               IF nVal = 1 THEN
                  LET cLetras = cLetras CLIPPED, ' MILLON '
               ELSE
                  LET cLetras = cLetras CLIPPED, ' MILLONES '
               END IF
            END IF
         END IF
      END IF

   END FOR

   IF cLetras[1,1] = ' ' THEN
      LET cLetras = cLetras[2,90]
   END IF

   IF nNumero = 1000000 THEN
      LET cLetras = cLetras CLIPPED,' DE '
   END IF

   LET cLetras = '(',cLetras CLIPPED,' PESOS ',cNumero[11,12],'/100 M.N.)'

   RETURN cLetras

END FUNCTION
################################################################################
```

- **CÁLCULO DEL TIEMPO EXTRA:**

Este procedimiento incluye **Abstracción de datos**, **Abstracción procedimental**, **Arquitectura** y **Modularidad** del programa para calcular el tiempo extra y guardarlo para su posterior aplicación en una quincena.

La ley en la materia dice:

TIEMPO EXTRAORDINARIO. MECANISMO DE CÁLCULO PARA SU PAGO CONFORME A LOS ARTÍCULOS 66 A 68 DE LA LEY FEDERAL DEL TRABAJO.

El artículo 66 de la Ley Federal del Trabajo establece que el tiempo extraordinario no podrá exceder de tres horas diarias ni de tres veces a la semana. Por otra parte, los numerales 67 y 68 de la citada ley señalan, en cuanto a su pago, que las horas extras que no rebasen ese límite se cubrirán con un 100% más del salario que corresponda a las horas de la jornada, mientras que las horas que excedan de nueve a la semana deberán pagarse con un 200% más del salario respectivo. Ahora bien, de dichos dispositivos se advierte un mecanismo para el cálculo de su pago basado no sólo en el máximo de nueve horas generadas en una semana, sino también por día, razón por la cual deberá atenderse a las horas realmente laboradas por cada día. En ese sentido, si un trabajador prestó sus servicios toda una semana generando dos horas extras diarias, es claro que las primeras seis horas extras originadas en los primeros tres días serán pagadas con un 100% más del salario, mientras que las restantes seis horas de los siguientes tres días con un 200% más.

TERCER TRIBUNAL COLEGIADO EN MATERIA DE TRABAJO DEL PRIMER CIRCUITO.

La interpretación "tradicional" otorgada a los artículos 66, 67 y 68 de la Ley Laboral hasta antes de la emisión del criterio jurisprudencial antes citado consistía en otorgar un valor adicional del 100% a las primeras nueve horas extras laboradas durante la semana, así como un valor adicional del 200% a las horas extraordinarias posteriores al límite semanal antes indicado. Por otro lado, en concordancia con la nueva interpretación que a los preceptos legales antes precisados realiza el Tercer Tribunal Colegiado del Primer Circuito, el cálculo de tiempo extraordinario no solo debe atender al total de horas laboradas durante la semana, pues además, ahora deben ser considerados también el número de días en que las mismas fueron laboradas, (pues a partir del cuarto día de la semana en que sea laborado tiempo extraordinario, este se considerará como triple para efectos de su pago), así como el número de horas extraordinarias laboradas por día, (pues al exceder tres horas de tiempo extraordinario en un día, dicho excedente deberá ser también considerado como triple para efectos de su pago).

Además de la interpretación anterior se debe tomar en cuenta que el tiempo extra se debe calcular por semana de lunes a domingo y esta puede ser pagada en dos quincenas, es decir, una parte en una quincena y la otra parte en otra quincena subsiguiente, de tal manera que se debe tomar en cuenta las horas pagadas en una quincena, ya sean las dobles y/o triples para calcular el tipo de horas extras que faltan por pagar en la siguiente quincena.

Para el cálculo del impuesto se deberá tomar la base gravable del tiempo extra y ésta se calcula de la siguiente manera:

BASE GRAVABLE Y EXENTA DE LAS HORAS EXTRA (2010)

La ley del ISR artículo 109 fracción primera considera que las horas extras de salario mínimo que no excedan los límites de la ley federal del trabajo deben ser exentas en su totalidad. Las horas de salarios superiores al mínimo deben ser consideradas el 50% exento y el 50% gravado; y el 50% exento no deberá exceder 5 veces el salario mínimo en la semana.

De acuerdo con el artículo 27 de la ley del seguro social se considera que las horas extras que no excedan los límites del la Ley Federal del Trabajo no integran el Salario Base de Cotización.

Por ejemplo (SM 2010 $57.46)

Trabajador: X

Salario Mensual	*Días Mes Ley Federal del Trabajo*	*Salario diario*	*No. Hrs. Extras jornada diurna*	*Salario por hora*	*Salario por hora extra 100%*	*Salario por hora extra 200%*
$12,000.00	*30*	*$400.00*	*8*	*$50.00*	*$100.00*	*$150.00*

Como ya es sabido, las horas extras se pueden calcular de dos formas, según la Ley Federal del Trabajo y según la costumbre.

Para mayor explicación y mejor comprensión se presenta de las dos formas a continuación:

Caso práctico

Caso Según Ley Federal del Trabajo	*Lunes*	*Martes*	*Miércoles*	*Suma*	*Exento p/ ISR*	*Gravado p/ ISR*
Cálculo LFT						
Hrs Extras laboradas	*4*	*3*	*2*	*9*	*5VSM*	
100% mas por hora	*$ 100.00*	*$ 100.00*	*$ 100.00*			
Total Horas extras	*$ 400.00*	*$ 300.00*	*$ 200.00*	*$ 900.00*		
Exención ISR						
Monto de horas que exceden el limite de LFT	*$ 100.00*			*$ 100.00*		*$ 100.00*
50% gravado del monto excede LTF	*$ 150.00*	*$ 150.00*	*$ 100.00*	*$ 400.00*		*$ 400.00*
50% Sujeto a no exceder 5VSM	*$ 150.00*	*$ 150.00*	*$ 100.00*	*$ 400.00*	*$ 287.30*	*$ 112.70*
SUMAS	*$ 400.00*	*$ 300.00*	*$ 200.00*	*$ 900.00*	*$ 287.30*	*$ 612.70*

Como se puede apreciar, el procedimiento de llenado, es fácil, ubicando en cada uno de los datos.

Se puede apreciar en "Exento para ISR" que dice 5 Veces el Salario Mínimo, lo cual se tiene que recordar, como viene señalado en la tabla de las percepciones que se gravan, Hacienda da un margen para reducir la base para el cálculo del impuesto. Es por eso la señalación de 5VSM.

Caso Según Costumbre	*Lunes*	*Martes*	*Miércoles*	*Suma*	*Exento p/ ISR*	*Gravado p/ ISR*
Cálculo LFT						
Hrs Extras laboradas	*4*	*3*	*2*	*9*	*5VSM*	
Hrs de 100% más	*3*	*3*	*2*			
100% mas por hora	*$ 100.00*	*$ 100.00*	*$ 100.00*			
Total 100% mas	*$ 300.00*	*$ 300.00*	*$ 200.00*			
Horas de 200% más	*1*					
200% mas por hora	*$ 150.00*					
Total hrs 200% mas	*$ 150.00*					
Total horas extras	*$ 450.00*	*$ 300.00*	*$ 200.00*	*$ 950.00*		
Exención ISR						
Monto de horas que exceden el limite de LFT	*$ 150.00*			*$ 150.00*		*$ 150.00*
50% gravado del monto excede LTF	*$ 150.00*	*$ 150.00*	*$ 100.00*	*$ 400.00*		*$ 400.00*
50% Sujeto a no exceder 5VSM	*$ 150.00*	*$ 150.00*	*$ 100.00*	*$ 400.00*	*$ 287.30*	*$ 112.70*
SUMAS	*$ 450.00*	*$ 300.00*	*$ 200.00*	*$ 950.00*	*$ 287.30*	*$ 662.70*

En este caso en particular, el método que se utilizará para el cálculo de las partes Exenta y Gravada será según Costumbre. Al igual que el pago de las horas extras en dos quincenas, se debe de tomar en cuenta si en la primera se aplicó todo el exento o queda algo por aplicarse en la siguiente quincena, y hay que llevar un control para esto también.

Asimismo, también existe el pago de días de descanso, los cuales también están contemplados en la Ley Federal del Trabajo y se calculan como sigue:

DÍAS DE DESCANSO TRIPES

Artículo 73.- Los trabajadores no están obligados a prestar servicios en sus días de descanso. Si se quebranta esta disposición, el patrón pagará al trabajador, independientemente del salario que le corresponda por el descanso, un salario doble por el servicio prestado.

Concepto	***Cantidad***	***Importe***
Salario diario		*$ 100.00*
Días trabajados	*7*	*$ 700.00*
Séptimo día	*Salario diario * 2*	*$ 200.00*
Total semanal		*$ 900.00*

Para el caso que nos ocupa, puede haber hasta 5 días de descanso.

De igual manera los días de descanso tienen una parte gravada y otra exenta y se calcula sumando el monto de los días de descanso al monto de las horas extra como parte de éstas y de ahí se obtienen las partes gravadas y exentas.

Y por último, también se paga una prima dominical por laborar a aquellos trabajadores que presten sus servicios en día domingo de acuerdo a Ley Federal del Trabajo.

PRIMA DOMINICAL

Artículo 71 - En los reglamentos de esta Ley se procurará que el día de descanso semanal sea el domingo. Los trabajadores que presten servicio en día domingo tendrán derecho a una prima adicional de un veinticinco por ciento, por lo menos, sobre el salario de los días ordinarios de trabajo.

También esta prima incluye una parte base gravable y una parte exenta que se calcula de acuerdo al siguiente artículo de la Ley Ingresos Sobre la Renta.

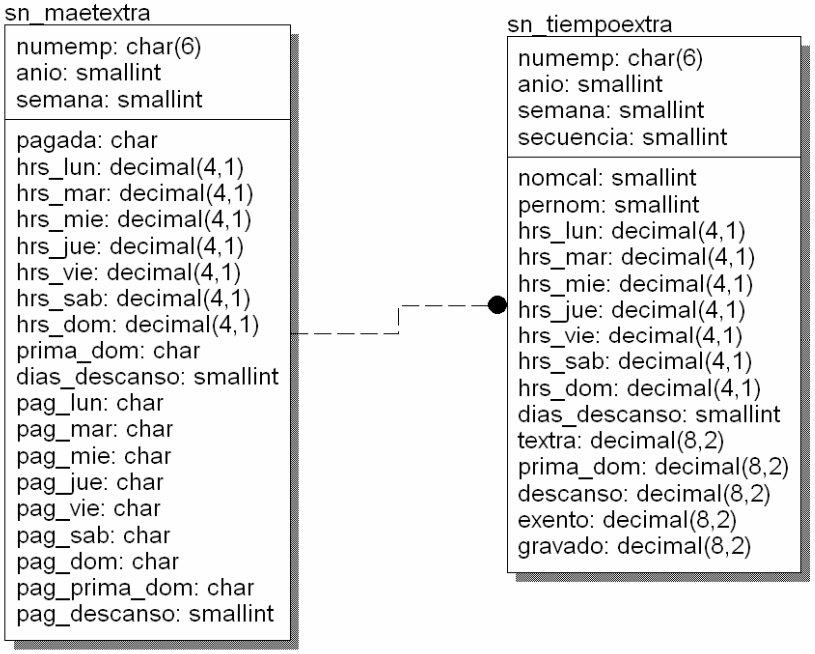
Art 109, Fracción XI - Las gratificaciones que reciban los trabajadores de sus patrones, durante un año de calendario, hasta el equivalente del salario mínimo general del área geográfica del trabajador elevado a 30 días, cuando dichas gratificaciones se otorguen en forma general; así como las primas vacacionales que otorguen los patrones durante el año de calendario a sus trabajadores en forma general y la participación de los trabajadores en las utilidades de las empresas, hasta por el equivalente a 15 días de salario mínimo general del área geográfica del trabajador, por cada uno de los conceptos señalados. ***Tratándose de primas dominicales hasta por el equivalente de un salario mínimo general del área geográfica del trabajador por cada domingo que se labore****.*

De acuerdo al planteamiento para el cálculo tanto del tiempo extra, días de descanso y prima dominical, así como de sus partes gravadas y exentas, siendo que se pueden pagar en una o más quincenas, que en una quincena pueden pagarse varias semanas, que se deberá controlar que conceptos se pagan en cada quincena, además de

generar un historial de éstos para futuras consultas, se llegó al siguiente diseño en sus partes que lo conforman:

Abstracción de datos

Se requiere construir las tablas en donde residan los datos por semana para cada trabajador que labore durante ese periodo tanto tiempo extra como días de descanso, indicando por día de la semana cuántas horas extras se laboraron, cuántos días de descanso y si también se le paga prima dominical. Dado que se pueden pagar estos conceptos en diferentes quincenas se tendrá que identificar en que quincena fue pagada cada parte, por lo cual se llegó al diseño de las siguientes tablas:



Se consideró la creación de un par de tablas maestra y esclava que contengan los datos requeridos.

La tabla maestra `sn_maetextra` contiene los totales de horas a pagarse por día de la semana, la cantidad de días descanso y si se paga prima dominical; además contiene columnas que indican cuáles días de tiempo extra fueron pagados, cuántos de descanso y si se pagó la prima dominical, además con la columna `pagada` que indica si esta semana ya fue pagada o no, independientemente de si fue un pago parcial. Así mismo contiene las columnas que identifican al trabajador, el año y semana correspondiente.

La tabla esclava `sn_tiempoextra` contiene los datos de cada ocurrencia parcial que pueda existir en la captura de los datos dada por la columna `secuencia` en cuyo registro contendrá las horas de tiempo extra registradas por día de la semana, los días de descanso, así como los importes que son resultado de los cálculos correspondientes por estos conceptos, tanto por su importe a pagar, como por las partes gravada y exenta. Así mismo contiene las columnas que identifican al trabajador, año, semana, ocurrencia, y nómina de cálculo y periodo en el que se pagó.

Abstracción procedimental

El procedimiento a seguir para el cálculo del tiempo extra, días de descanso y prima dominical es el siguiente:

1. El sueldo diario se calcula dividiendo el sueldo mensual del empleado entre 30.
2. El sueldo por hora se calcula dividiendo el sueldo diario entre 7.

3. Si ya hubo un pago previo de tiempo extra correspondiente a la semana en cuestión se obtienen los valores de las horas dobles pagadas, horas triples pagadas, días de descanso pagados, si se pagó prima dominical, los importes de tiempo extra, prima dominical, días de descanso, al igual que los montos gravados y exentos.
4. Se van sumando las horas de cada día de lunes a domingo, contando sí hay más de 3 días con tiempo extra, si lo hay se suman las horas de tiempo extra de los días excedentes a éstos a las triples, y al mismo tiempo se va revisando si en cada día hay más de 3 horas, si las hay se suman 3 a las dobles y las excedentes a estas se suman a las triples, de lo contrario se suman a las dobles.
5. A las horas dobles y triples se le restan las horas dobles y triples pagadas con anterioridad respectivamente.
6. El importe de horas extras a pagar se calcula multiplicando por dos las horas dobles por el sueldo por hora más la multiplicación de las horas triples por tres por el sueldo por hora.
7. El importe de días de descanso se calcula multiplicando el sueldo diario por dos por la diferencia del número de días de descanso menos los días de descanso pagados.
8. Para calcular el gravado y el exento del tiempo extra y días de descanso se compara la suma de los importes de tiempo extra más el tiempo extra anterior más los días de descanso más los días de descanso anterior y dividiendo todo esto entre dos, con cinco veces el salario mínimo. Si es mayor, se compara el exento aplicado anteriormente con cinco veces el salario mínimo, si es menor se aplica la diferencia de cinco salarios mínimos menos el exento anterior, de lo contrario será cero; y el gravado será la suma de los importes de tiempo extra más los días de descanso menos el exento. Si la suma de los importes de tiempo extra más el tiempo extra anterior más los días de descanso más los días de descanso anterior y dividiendo todo esto entre dos es menor o igual que cinco veces el salario mínimo, el exento será la operación anterior menos el exento anterior; y el gravado será la suma de los importes de tiempo extra más los días de descanso menos el exento.
9. Para el cálculo de la prima dominical, su exento y gravado, si se debe de pagar y no se ha pagado previamente, la prima equivale al veinticinco por ciento del sueldo diario, si la prima es mayor que un salario mínimo, el exento será la suma del exento del tiempo extra y días de descanso calculados en el punto anterior más un salario mínimo y la parte gravada la suma del gravado del tiempo extra y días de descanso calculados en el punto anterior más la prima dominical menos un salario mínimo; si resulta ser menor o igual, el exento será suma del exento del tiempo extra y días de descanso calculados en el punto anterior más la prima dominical.

Los datos serán almacenados en las tablas definidas en la Abstracción de datos y serán explotados tanto dentro de este procedimiento, así como en el de la inicialización de un nuevo periodo de nómina y en el cálculo del impuesto. Si una parte de una semana de tiempo extra ya fue pagada en una nómina, ésta no podrá ser modificada y se tendrá que agregar una nueva ocurrencia de tiempo extra para la semana correspondiente, la cual estará disponible hasta que se incluya y sea pagada en otra quincena.

Arquitectura

El sistema está organizado en diferentes componentes, los cuales cada uno es un elemento independiente y tiene su propia función, por lo cual siguiendo esta arquitectura, este proceso será igualmente desarrollado en un componente independiente.

Modularidad

El patrón del diseño de los componentes (programas) de captura de datos estará dividido en diferentes módulos que interactúen entre sí para cumplir con el objetivo requerido. Los módulos estarán divididos en una parte de declaración de variables globales, una función principal para inicializar variables y llamar a la función de menú de opciones de Altas, Bajas, Cambios, Encuentra, Detalle, Siguiente, Previo, Reporte y Fin, para controlar las diferentes funciones del programa y lograr su objetivo. Además contará con funciones para buscar y traer los datos de las tablas, desplegarlos en la pantalla y validar los datos introducidos.

- **CÁLCULO DEL IMPUESTO:**

Este procedimiento incluye **Abstracción procedimental**, **Arquitectura** e **Independencia funcional** para calcular el impuesto.

El **ISR** tiene como función gravar los ingresos y el salario percibido por un trabajador. Este impuesto se calcula con la tarifa establecida en el artículo 96 de la **Ley del Impuesto Sobre la Renta (LISR)**, la cual se cita a continuación: ***"La retención se calculará aplicando la siguiente tabla a la totalidad de los ingresos obtenidos en un mes de calendario"***:

Este impuesto se calcula, por periodos, semanales, catorcenales, quincenales o mensuales, todo depende de los periodos de pago de la empresa. Sin embargo, la forma mas exacta para cálculo del ISR, es hacerlo mensualmente.

TARIFA APLICABLE A 2011			
Límite inferior	***Límite superior***	***Cuota fija***	***Porciento para aplicarse sobre el excedente del límite inferior***
$	***$***	***$***	***%***
0.01	*496.07*	*0.00*	*1.92*
496.08	*4,210.41*	*9.52*	*6.40*
4,210.42	***7,399.42***	***247.23***	***10.88***
7,399.43	*8,601.50*	*594.24*	*16.00*
8,601.51	*10,298.35*	*786.55*	*17.92*
10,298.36	*20,770.29*	*1,090.62*	*21.36*
20,770.30	*32,736.83*	*3,327.42*	*23.52*
32,736.84	*En adelante*	*6,141.95*	*30.00*

Cálculo del ISR mensual*:*

1. *Se ubica el ingreso gravable, entre el **Límite inferior** y **Límite superior**.*
2. *Luego se le resta al ingreso gravable, el **Límite inferior**.*
3. *Después se multiplica el resultado del punto anterior por la tasa de la columna **Porciento para aplicarse sobre el excedente del limite inferior**.*
4. *Al resultado del punto anterior, se le suma el importe de la columna **Cuota fija** y con esta serie de operaciones se obtiene el **ISR**.*

Por ejemplo, trabajador X ganando $5,000.00 mensuales

Operaciones:

5,000.00	
Menos Límite Inferior	*4,210.42*
Excedente	*789.58*
% sobre excedente	*10.88*
Subtotal	*85.91*
Cuota Fija	*247.23*
ISR	***333.14***

La ley permite aplicar la tabla del subsidio al empleo, a los trabajadores asalariados, por lo que tomando como base la tabla mensual de subsidio al empleo:

TABLA DEL SUBSIDIO PARA EL EMPLEO APLICABLE A LA TARIFA DEL NUMERAL 5 DEL RUBRO B		
Límite inferior	***Límite superior***	***Cantidad de subsidio para el empleo mensual***
$	***$***	***$***
0.01	*1,768.96*	*407.02*
1,768.97	*2,653.38*	*406.83*
2,653.39	*3,472.84*	*406.62*
3,472.85	*3,537.87*	*392.77*
3,537.88	*4,446.15*	*382.46*
4,446.16	*4,717.18*	*354.23*
4,717.19	***5,335.42***	***324.87***
5,335.43	*6,224.67*	*294.63*
6,224.68	*7,113.90*	*253.54*
7,113.91	*7,382.33*	*217.61*
7,382.34	*En adelante*	*0*

Cálculo del subsidio al empleo a restar al resultado del calculo del ISR 2011

1. *Se ubica el ingreso gravable (5,000.00 pesos) entre el **Límite inferior** y el **Límite superior** de la tabla, y la tercera columna indica el subsidio mensual (324.87)*
2. *Se resta al ISR determinado, y se tiene el **ISR neto a retener***

ISR	*333.14*
Subsidio al empleo	*324.87*
ISR a retener	***8.27***

En este caso en particular el cálculo debe de hacerse por quincena.

*Para el **Cálculo del ISR en la primera quincena** del mes se siguen los siguientes pasos:*

1. *Se obtiene el **Impuesto Normal Mensual** (ISR a retener anterior) con la suma de todos los conceptos gravables de la quincena por 2, sin considerar las percepciones y deducciones adicionales, a través del procedimiento del **cálculo del ISR mensual** anterior.*
2. *Se obtiene el **Impuesto Total Mensual** con la suma de todos los conceptos gravables de la quincena más los conceptos gravables de la quincena sin considerar las percepciones y deducciones adicionales (se supone la segunda quincena sin percepciones ni deducciones adicionales), a través del procedimiento del **cálculo del ISR mensual**.*
3. *Se obtiene el **Impuesto de la primera quincena** restando del **Impuesto Total Mensual** la **mitad** del **Impuesto Normal Mensual**.*

*Para el **Cálculo del ISR en la segunda quincena** del mes se siguen los siguientes pasos:*

1. *Se obtiene el **Impuesto de la primera quincena** sumando el ISR ya calculado y guardado de la nómina correspondiente.*
2. *Se obtiene la **Percepción Mensual** sumando todos los conceptos gravables ambas quincenas.*
3. *Se obtiene el **Impuesto Total Mensual** con la suma de todos los conceptos gravables tanto de la quincena anterior como de la actual, al igual con las faltas de ambas quincenas.*
4. *Se obtiene el **Impuesto de la segunda quincena** restando del **Impuesto Total Mensual** el **Impuesto de la primera quincena**.*

Abstracción procedimental

El procedimiento a seguir para el cálculo del impuesto (ISR) es el siguiente:

1. Llamaremos Base Impuesto a la suma de conceptos que son aquellos que se gravan y participan en el cálculo del impuesto.
2. Se busca la Base Impuesto en la tabla de ISPT para determinar el rango en el que se encuentre, obteniendo el Límite Inferior, Cuota Fija y el Porcentaje Excedente de Límite Inferior.
3. Se obtiene el Excedente del Límite Inferior restando del Base Impuesto el Límite Inferior.
4. Se obtiene el Impuesto Marginal multiplicando el Excedente del Límite Inferior por el Porcentaje Excedente de Límite Inferior.
5. Se obtiene el Impuesto Total sumando el Impuesto Marginal con la Cuota Fija.
6. Se obtiene la Diferencia restando del Impuesto Total el Subsidio Acreditado.
7. Se busca el Base Impuesto en la tabla de Subsidio para determinar el rango en el que se encuentre, obteniendo el Crédito al Salario.
8. Se obtiene el Factor de Días Laborados dividiendo la diferencia de 30 menos las faltas entre 30.
9. Se recalcula el Crédito al Salario multiplicándolo por el Factor de Días Laborados.
10. Finalmente se obtiene el ISR restando de la Diferencia el Crédito al Salario.

Para el Cálculo del ISR en la primera quincena del mes se siguen los siguientes pasos:

1. Se obtiene el Impuesto Normal Mensual con la suma de todos los conceptos gravables de la quincena por 2, sin considerar las percepciones y deducciones adicionales, a través del procedimiento Impuesto Mensual.
2. Se obtiene el Impuesto Total Mensual con la suman de todos los conceptos gravables de la quincena más los conceptos gravables de la quincena sin considerar las percepciones y deducciones adicionales (se supone la segunda quincena normal), y se toman en cuenta las faltas de la quincena, a través del procedimiento Impuesto Mensual.
3. Se obtiene el Impuesto de la Primera Quincena restando del Impuesto Total Mensual la mitad del Impuesto Normal Mensual.

Para el Cálculo del ISR en la segunda quincena del mes se siguen los siguientes pasos:

1. Se obtiene el Impuesto de la Primera Quincena sumando el ISR de todas las nóminas de la primera quincena.
2. Se obtiene la Percepción Mensual sumando todos los conceptos gravables tanto de la quincena anterior como de la actual, al igual que las faltas de ambas quincenas.

3. Se obtiene el Impuesto Total Mensual con la suma de todos los conceptos gravables tanto de la quincena anterior como de la actual, al igual con las faltas de ambas quincenas.
4. Se obtiene el Impuesto de la Segunda Quincena restando del Impuesto Total Mensual el Impuesto de la Primera Quincena.

Arquitectura

El sistema está organizado en diferentes componentes, los cuales cada uno es un elemento independiente y tiene su propia función, por lo cual siguiendo esta arquitectura, este proceso será igualmente desarrollado en un componente independiente.

Independencia funcional

Se decidió desarrollar un par de Procedimientos Almacenados (SPL – Store Procedure Language) que van insertos en la Base de Datos para el cálculo del impuesto, puesto que serán llamados desde diferentes módulos del sistema, incluso desde consultas no planeadas.

El procedimiento (SPL) que efectúa el cálculo del impuesto mensual tendrá una cohesión alta puesto que solo calculará el impuesto, mientras que otro llamará a éste tantas veces como se requiera para hacer los cálculos por cada una de las dos quincenas del mes correspondiente. El acoplamiento entre el segundo y el primero es baja, requiriendo cinco argumentos y devolviendo al primero un solo valor (el ISR mensual).

11 ESPECIFICACIÓN Y DISEÑO DE PROCESOS

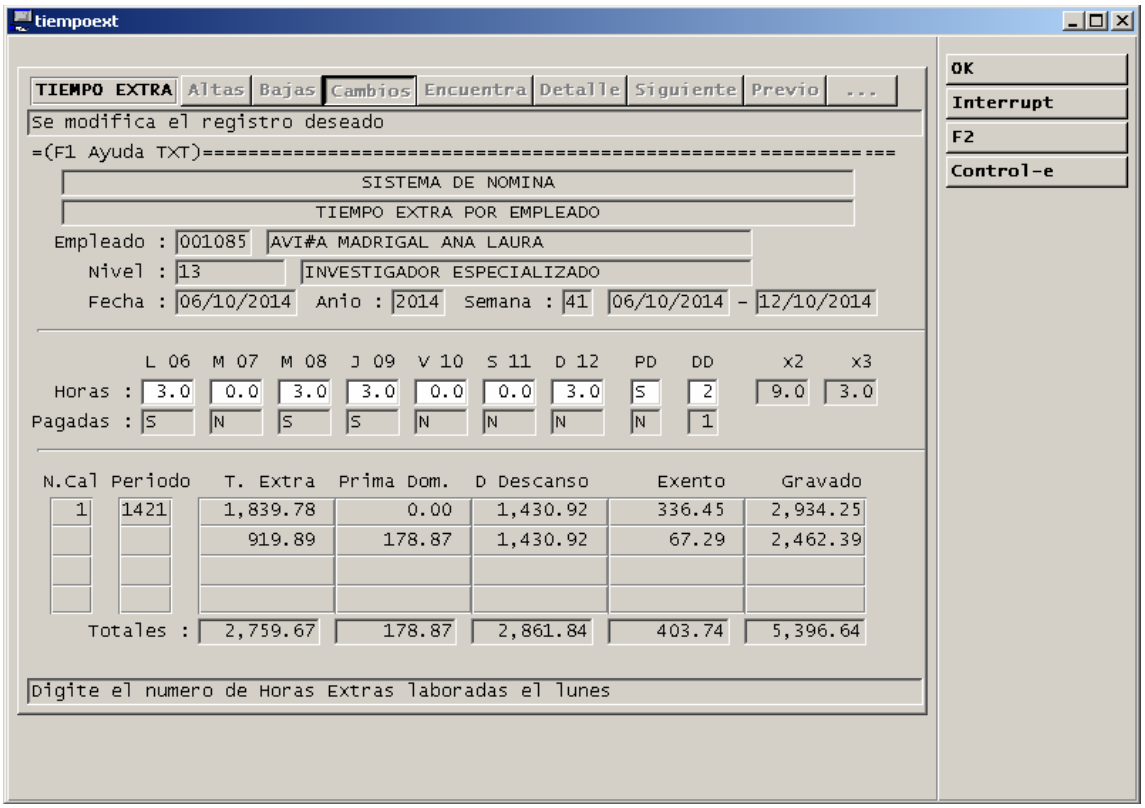
A continuación se presentan las especificaciones y diseño de procesos de dos de los procesos requeridos del sistema vistos anteriormente: **Cálculo del tiempo extra** y **Cálculo del impuesto**.

- **CÁLCULO DEL TIEMPO EXTRA:**

  En este caso en particular las funciones de Altas y Cambios llamarán a la función medular del programa, el cálculo del tiempo extra. Esta función utilizará las variables globales del programa debido a que son muchas las variables que intervienen en el cálculo como para lograr en ésta una independencia funcional, además de ser una función llamada cada vez que se captura o modifica un valor del tiempo extra; y contará con la llamada a otra función para mostrar en la pantalla los valores calculados interactivamente.

  El procedimiento en conjunto debe de tomar en cuenta si ya está pagada parte de una semana de tiempo extra y crear un nuevo registro para agregar la parte faltante de la semana que se pagará en la siguiente quincena y determinar si todavía se pagarán horas extras dobles o si ya serán calculadas triples y si aún quedan importes exentos por aplicar.

  A continuación se muestra la interfaz y la función del programa que capturará los datos en este procedimiento:




```
################################################################################
FUNCTION altas(vban)
   DEFINE
      vr                RECORD LIKE sn_maetextra.*,
      cNombreNomina     CHAR(30),
      cNombreCalculo    CHAR(50),
      cNombrePeriodo    CHAR(50),
      cNombreEmpleado   CHAR(60),
      vban,
      verr              SMALLINT,
      vdesc             CHAR(60)

   LET vdesc = NULL
-- CALL mensa("A")

   IF vban = 1 THEN
      IF vg_cont2 = 0 THEN
         ERROR "Entrar primero a la opcion ENCUENTRA"
         RETURN
      END IF
      If vgentra = 1 THEN
         CALL w_men("El registro ya ha sido borrado")
         RETURN
      END IF
   ELSE
      LET vr.* = vgrtabla.*
--    CALL inicializa()
      CLEAR FORM
      CALL borra()
   END If

   OPTIONS
      MESSAGE LINE 24


   FOR i = 1 TO 7
      LET j = (7 + 6*i)
      DISPLAY '  ' AT 10,j
   END FOR

   LET dFecha = ''

   LET vgrtabla.hrs_lun = 0
   LET vgrtabla.hrs_mar = 0
   LET vgrtabla.hrs_mie = 0
   LET vgrtabla.hrs_jue = 0
   LET vgrtabla.hrs_vie = 0
   LET vgrtabla.hrs_sab = 0
```

```
LET vgrtabla.hrs_dom = 0
LET vgrtabla.prima_dom = 'N'
LET vgrtabla.dias_descanso = 0

FOR vg_cont4 = 1 TO 20
   INITIALIZE arr_tiempoextra[vg_cont4].* TO NULL
   LET arr_tiempoextra[vg_cont4].textra    = 0
   LET arr_tiempoextra[vg_cont4].prima_dom = 0
   LET arr_tiempoextra[vg_cont4].descanso  = 0
   LET arr_tiempoextra[vg_cont4].exento    = 0
   LET arr_tiempoextra[vg_cont4].gravado   = 0
END FOR

DISPLAY arr_tiempoextra[1].* TO scr_tiempoextra[1].*

LET r_sn_tiempoextra.secuencia = 1

LET vgrtabla.pag_lun       = 'N'
LET vgrtabla.pag_mar       = 'N'
LET vgrtabla.pag_mie       = 'N'
LET vgrtabla.pag_jue       = 'N'
LET vgrtabla.pag_vie       = 'N'
LET vgrtabla.pag_sab       = 'N'
LET vgrtabla.pag_dom       = 'N'
LET vgrtabla.pag_prima_dom = 'N'
LET vgrtabla.pag_descanso  = 0

LET nAntHoras2   = 0
LET nAntHoras3   = 0
LET nAntTextra   = 0
LET nAntPrima    = 0
LET nAntDescanso = 0
LET nAntExento   = 0
LET nAntGravado  = 0


DISPLAY 0, 0, 0, 0 TO nTotTextra, nTotPrima, nTotExento, nTotGravado


INPUT vgrtabla.numemp,  dFecha, vgrtabla.anio, vgrtabla.semana,
      vgrtabla.hrs_lun, vgrtabla.hrs_mar,
      vgrtabla.hrs_mie, vgrtabla.hrs_jue,
      vgrtabla.hrs_vie, vgrtabla.hrs_sab,
      vgrtabla.hrs_dom, vgrtabla.prima_dom, vgrtabla.dias_descanso
   WITHOUT DEFAULTS
   FROM sn_maetextra.numemp,  dFecha, sn_maetextra.anio, sn_maetextra.semana,
        sn_maetextra.hrs_lun, sn_maetextra.hrs_mar,
        sn_maetextra.hrs_mie, sn_maetextra.hrs_jue,
        sn_maetextra.hrs_vie, sn_maetextra.hrs_sab,
        sn_maetextra.hrs_dom, sn_maetextra.prima_dom, sn_maetextra.dias_descanso

   BEFORE INPUT
      CALL pon_zflag(0,0)

   BEFORE FIELD numemp
      CALL pon_zflag(0,0)

   AFTER FIELD numemp
      IF vgrtabla.numemp IS NULL THEN
         ERROR " El Numero de Empleado debe ser capturado "
         DISPLAY "" TO cNombreEmpleado
         NEXT FIELD numemp
      ELSE
         LET cNombreEmpleado = nombre_empleado(vgrtabla.numemp)
         IF cNombreEmpleado IS NULL OR LENGTH(cNombreEmpleado) = 0 THEN
            ERROR "No Existe Empleado en la Nomina 11 o esta dado de Baja"
            NEXT FIELD numemp
         ELSE
            DISPLAY BY NAME cNombreEmpleado

            SELECT c01_clapag, c01_descla, d01_impcon / 30
               INTO cCveNivel, cNomNivel, nSueldoDiario
               FROM m301_clpg
               WHERE c01_tipnom = '11' AND
                     c01_clapag = (SELECT c20_cpsn
                                   FROM m320_memp
                                   WHERE c20_numemp = vgrtabla.numemp)

            LET nSueldoHora = nSueldoDiario / 7

            DISPLAY BY NAME cCveNivel, cNomNivel
         END IF
      END IF
      CALL pon_zflag(0,0)

   AFTER FIELD dFecha
      IF dFecha IS NOT NULL THEN
         IF DAY(dFecha) = 1 AND MONTH(dFecha) = 1 AND WEEKDAY(dFecha) = 0 THEN
            LET dFecha = dFecha - 1
            DISPLAY BY NAME dFecha
         END IF
         LET vgrtabla.semana = busca_semana(dFecha)
         LET vgrtabla.anio = YEAR(dFecha)
         DISPLAY BY NAME vgrtabla.semana, vgrtabla.anio
      END IF
      CALL pon_zflag(0,0)
```

```
AFTER FIELD anio
   IF vgrtabla.anio IS NULL THEN
      ERROR " El Anio debe ser capturado "
      NEXT FIELD anio
   ELSE
      IF vgrtabla.anio > YEAR(TODAY) OR (vgrtabla.anio < YEAR(TODAY)-1) THEN
         ERROR " Anio fuera de rango, debe ser el actual o el anterior "
         NEXT FIELD anio
      END IF
   END IF
   CALL pon_zflag(0,0)

AFTER FIELD semana
   IF vgrtabla.semana IS NULL THEN
      ERROR " El Numero de la Semana debe ser capturada "
      NEXT FIELD semana
   ELSE
      IF vgrtabla.semana < 1 OR vgrtabla.semana > 53 THEN
         ERROR " Semana fuera de rango (1-53) "
         NEXT FIELD semana
      ELSE
         CALL des_fechasemana(vgrtabla.anio, vgrtabla.semana)
         LET dFecha = dFechaIni
         DISPLAY dFecha, dFechaIni, dFechaFin TO
                 FORMONLY.dFecha, FORMONLY.cFechaIni, FORMONLY.cFechaFin
         CALL despl_dias(dFechaIni)

         SELECT *
            FROM sn_maetextra
            WHERE numemp = vgrtabla.numemp AND
                  anio   = vgrtabla.anio   AND
                  semana = vgrtabla.semana

         IF STATUS != NOTFOUND THEN
            ERROR "Ya existe este registro, entre a la opción Encuentra y Cambios"
            NEXT FIELD semana
         END IF

      END IF
   END IF
   CALL pon_zflag(0,0)

AFTER FIELD sn_maetextra.hrs_lun
   IF vgrtabla.hrs_lun IS NULL THEN
      ERROR " El Numero de horas debe ser capturada "
      NEXT FIELD sn_maetextra.hrs_lun
   ELSE
      IF vgrtabla.hrs_lun < 0 OR vgrtabla.hrs_lun > 17 THEN
         ERROR " Numero de horas fuera de rango (0-17)"
         NEXT FIELD sn_maetextra.hrs_lun
      ELSE
         CALL cal_tiempoextra()
      END IF
   END IF

AFTER FIELD sn_maetextra.hrs_mar
   IF vgrtabla.hrs_mar IS NULL THEN
      ERROR " El Numero de horas debe ser capturada "
      NEXT FIELD sn_maetextra.hrs_mar
   ELSE
      IF vgrtabla.hrs_mar < 0 OR vgrtabla.hrs_mar > 17 THEN
         ERROR " Numero de horas fuera de rango (0-17)"
         NEXT FIELD sn_maetextra.hrs_mar
      ELSE
         CALL cal_tiempoextra()
      END IF
   END IF

AFTER FIELD sn_maetextra.hrs_mie
   IF vgrtabla.hrs_mie IS NULL THEN
      ERROR " El Numero de horas debe ser capturada "
      NEXT FIELD sn_maetextra.hrs_mie
   ELSE
      IF vgrtabla.hrs_mie < 0 OR vgrtabla.hrs_mie > 17 THEN
         ERROR " Numero de horas fuera de rango (0-17)"
         NEXT FIELD sn_maetextra.hrs_mie
      ELSE
         CALL cal_tiempoextra()
      END IF
   END IF

AFTER FIELD sn_maetextra.hrs_jue
   IF vgrtabla.hrs_jue IS NULL THEN
      ERROR " El Numero de horas debe ser capturada "
      NEXT FIELD sn_maetextra.hrs_jue
   ELSE
      IF vgrtabla.hrs_jue < 0 OR vgrtabla.hrs_jue > 17 THEN
         ERROR " Numero de horas fuera de rango (0-17)"
         NEXT FIELD sn_maetextra.hrs_jue
      ELSE
         CALL cal_tiempoextra()
      END IF
   END IF

AFTER FIELD sn_maetextra.hrs_vie
   IF vgrtabla.hrs_vie IS NULL THEN
      ERROR " El Numero de horas debe ser capturada "
```

```
      NEXT FIELD sn_maetextra.hrs_vie
   ELSE
      IF vgrtabla.hrs_vie < 0 OR vgrtabla.hrs_vie > 17 THEN
         ERROR " Numero de horas fuera de rango (0-17)"
         NEXT FIELD sn_maetextra.hrs_vie
      ELSE
         CALL cal_tiempoextra()
      END IF
   END IF

AFTER FIELD sn_maetextra.hrs_sab
   IF vgrtabla.hrs_sab IS NULL THEN
      ERROR " El Numero de horas debe ser capturada "
      NEXT FIELD sn_maetextra.hrs_sab
   ELSE
      IF vgrtabla.hrs_sab < 0 OR vgrtabla.hrs_sab > 24 THEN
         ERROR " Numero de horas fuera de rango (0-24)"
         NEXT FIELD sn_maetextra.hrs_sab
      ELSE
         CALL cal_tiempoextra()
      END IF
   END IF

AFTER FIELD sn_maetextra.hrs_dom
   IF vgrtabla.hrs_dom IS NULL THEN
      ERROR " El Numero de horas debe ser capturada "
      NEXT FIELD sn_maetextra.hrs_dom
   ELSE
      IF vgrtabla.hrs_dom < 0 OR vgrtabla.hrs_dom > 24 THEN
         ERROR " Numero de horas fuera de rango (0-24)"
         NEXT FIELD sn_maetextra.hrs_dom
      ELSE
         CALL cal_tiempoextra()
      END IF
   END IF

AFTER FIELD sn_maetextra.prima_dom
   IF vgrtabla.prima_dom IS NULL THEN
      ERROR " Prima Dominical debe ser capturada "
      NEXT FIELD sn_maetextra.prima_dom
   ELSE
      IF vgrtabla.prima_dom != 'S' AND vgrtabla.prima_dom != 'N' THEN
         ERROR " Prima Dominical fuera de rango (S/N)"
         NEXT FIELD sn_maetextra.prima_dom
      ELSE
         CALL cal_tiempoextra()
      END IF
   END IF

AFTER FIELD sn_maetextra.dias_descanso
   IF vgrtabla.dias_descanso IS NULL THEN
      ERROR " Dias de descabso debe ser capturado "
      NEXT FIELD sn_maetextra.dias_descanso
   ELSE
      IF vgrtabla.dias_descanso < 0 OR vgrtabla.dias_descanso > 7 THEN
         ERROR " Dias de descanso fuera de rango (0-7)"
         NEXT FIELD sn_maetextra.dias_descanso
      ELSE
         CALL cal_tiempoextra()
      END IF
   END IF

AFTER INPUT
   IF NOT INT_FLAG THEN
      IF vgrtabla.numemp IS NULL THEN
         ERROR " El Numero de Empleado debe ser capturado "
         DISPLAY "" TO cNombreEmpleado
         NEXT FIELD sn_maetextra.numemp
      ELSE
         LET cNombreEmpleado = nombre_empleado(vgrtabla.numemp)
         IF cNombreEmpleado IS NULL THEN
            ERROR "No Existe Empleado"
            NEXT FIELD sn_maetextra.numemp
         ELSE
            DISPLAY BY NAME cNombreEmpleado

            SELECT c01_clapag, c01_descla, d01_impcon / 30
               INTO cCveNivel, cNomNivel, nSueldoDiario
               FROM m301_clpg
               WHERE c01_tipnom = '11' AND
                     c01_clapag = (SELECT c20_cpsn
                                   FROM m320_memp
                                   WHERE c20_numemp = vgrtabla.numemp)

            LET nSueldoHora = nSueldoDiario / 7

            DISPLAY BY NAME cCveNivel, cNomNivel

            SELECT COUNT(*)
               INTO verr
               FROM sn_maetextra
               WHERE numemp = vgrtabla.numemp AND
                     anio   = vgrtabla.anio   AND
                     semana = vgrtabla.semana
```

```
IF verr != 0 THEN
   ERROR "Ya existe el registro "
   NEXT FIELD sn_maetextra.numemp
ELSE
   IF vgrtabla.anio IS NULL THEN
      ERROR " El Anio debe ser capturado "
      NEXT FIELD sn_maetextra.anio
   ELSE
      IF vgrtabla.anio > YEAR(TODAY) OR (vgrtabla.anio < YEAR(TODAY)-1) THEN
         ERROR " Anio fuera de rango, debe ser el actual o el anterior "
         NEXT FIELD sn_maetextra.anio
      ELSE
         IF vgrtabla.semana IS NULL THEN
            ERROR " El Numero de la Semana debe ser capturada "
            NEXT FIELD sn_maetextra.semana
         ELSE
            IF vgrtabla.semana < 1 OR vgrtabla.semana > 53 THEN
               ERROR " Semana fuera de rango (1-53) "
               NEXT FIELD sn_maetextra.semana
            ELSE
               CALL des_fechasemana(vgrtabla.anio, vgrtabla.semana)
               DISPLAY dFechaIni, dFechaFin TO FORMONLY.cFechaIni, FORMONLY.cFechaFin
               IF vgrtabla.hrs_lun IS NULL THEN
                  ERROR " El Numero de horas debe ser capturada "
                  NEXT FIELD sn_maetextra.hrs_lun
               ELSE
                  IF vgrtabla.hrs_lun < 0 OR vgrtabla.hrs_lun > 17 THEN
                     ERROR " Numero de horas fuera de rango (0-17)"
                     NEXT FIELD sn_maetextra.hrs_lun
                  END IF
               END IF
               IF vgrtabla.hrs_mar IS NULL THEN
                  ERROR " El Numero de horas debe ser capturada "
                  NEXT FIELD sn_maetextra.hrs_mar
               ELSE
                  IF vgrtabla.hrs_mar < 0 OR vgrtabla.hrs_mar > 17 THEN
                     ERROR " Numero de horas fuera de rango (0-17)"
                     NEXT FIELD sn_maetextra.hrs_mar
                  END IF
               END IF
               IF vgrtabla.hrs_mie IS NULL THEN
                  ERROR " El Numero de horas debe ser capturada "
                  NEXT FIELD sn_maetextra.hrs_mie
               ELSE
                  IF vgrtabla.hrs_mie < 0 OR vgrtabla.hrs_mie > 17 THEN
                     ERROR " Numero de horas fuera de rango (0-17)"
                     NEXT FIELD sn_maetextra.hrs_mie
                  END IF
               END IF
               IF vgrtabla.hrs_jue IS NULL THEN
                  ERROR " El Numero de horas debe ser capturada "
                  NEXT FIELD sn_maetextra.hrs_jue
               ELSE
                  IF vgrtabla.hrs_jue < 0 OR vgrtabla.hrs_jue > 17 THEN
                     ERROR " Numero de horas fuera de rango (0-17)"
                     NEXT FIELD sn_maetextra.hrs_jue
                  END IF
               END IF
               IF vgrtabla.hrs_vie IS NULL THEN
                  ERROR " El Numero de horas debe ser capturada "
                  NEXT FIELD sn_maetextra.hrs_vie
               ELSE
                  IF vgrtabla.hrs_vie < 0 OR vgrtabla.hrs_vie > 17 THEN
                     ERROR " Numero de horas fuera de rango (0-17)"
                     NEXT FIELD sn_maetextra.hrs_vie
                  END IF
               END IF
               IF vgrtabla.hrs_sab IS NULL THEN
                  ERROR " El Numero de horas debe ser capturada "
                  NEXT FIELD sn_maetextra.hrs_sab
               ELSE
                  IF vgrtabla.hrs_sab < 0 OR vgrtabla.hrs_sab > 24 THEN
                     ERROR " Numero de horas fuera de rango (0-24)"
                     NEXT FIELD sn_maetextra.hrs_sab
                  END IF
               END IF
               IF vgrtabla.hrs_dom IS NULL THEN
                  ERROR " El Numero de horas debe ser capturada "
                  NEXT FIELD sn_maetextra.hrs_dom
               ELSE
                  IF vgrtabla.hrs_dom < 0 OR vgrtabla.hrs_dom > 24 THEN
                     ERROR " Numero de horas fuera de rango (0-24)"
                     NEXT FIELD sn_maetextra.hrs_dom
                  END IF
               END IF
               IF vgrtabla.prima_dom IS NULL THEN
                  ERROR " Prima Dominical debe ser capturada "
                  NEXT FIELD sn_maetextra.prima_dom
               ELSE
                  IF vgrtabla.prima_dom != 'S' AND vgrtabla.prima_dom != 'N' THEN
                     ERROR " Prima Dominical fuera de rango (S/N)"
                     NEXT FIELD sn_maetextra.prima_dom
                  END IF
               END IF
               IF vgrtabla.dias_descanso IS NULL THEN
                  ERROR " Dias de descabso debe ser capturado "
                  NEXT FIELD sn_maetextra.dias_descanso
               ELSE
```

```
                                    IF vgrtabla.dias_descanso < 0 OR vgrtabla.dias_descanso > 7 THEN
                                       ERROR " Dias de descanso fuera de rango (0-7)"
                                       NEXT FIELD sn_maetextra.dias_descanso
                                    END IF
                                 END IF

                                 CALL cal_tiempoextra()

                              END IF
                           END IF
                        END IF
                     END IF
                  END IF
               END IF
            END IF

            CALL pon_zflag(0,0)

         ON KEY ("CONTROL-E")
            CALL Calendar()

         ON KEY (F2)
             CALL z_zoom()

      END INPUT

      IF INT_FLAG THEN
         LET vgrtabla.* = vr.*
         LET INT_FLAG = FALSE
         IF vban = 1 THEN # Si es cambios
            CALL w_men("Proceso de Cambios Abortado")
            CALL traer(0,0)
         ELSE
            CALL w_men("Proceso de Alta Abortado")
            IF vg_cont2 = 0 THEN
               CLEAR FORM
               CALL borra()
            ELSE
               CALL despliega(2)
            END IF
         END IF
      ELSE
         LET vgrtabla.pagada = 'N'

         INSERT INTO sn_maetextra
            VALUES (vgrtabla.*)

         INSERT INTO sn_tiempoextra
            VALUES (vgrtabla.numemp, vgrtabla.anio, vgrtabla.semana, 1, 0, 0,
                    vgrtabla.hrs_lun, vgrtabla.hrs_mar, vgrtabla.hrs_mie, vgrtabla.hrs_jue,
                    vgrtabla.hrs_vie, vgrtabla.hrs_sab, vgrtabla.hrs_dom, vgrtabla.dias_descanso,
                    arr_tiempoextra[1].textra, arr_tiempoextra[1].prima_dom, arr_tiempoextra[1].descanso,
                    arr_tiempoextra[1].exento, arr_tiempoextra[1].gravado)

         CALL SET_COUNT(0)
         CALL w_men("Alta Realizada")
      END IF

   END FUNCTION
   ###############################################################################
```

A continuación se muestra el código de la función que hace el cálculo del tiempo extra, prima dominical, días de descanso y sus respectivos exentos y gravados:

```
###############################################################################
FUNCTION cal_tiempoextra()
   DEFINE
      nIndice           SMALLINT


   LET nIndice = r_sn_tiempoextra.secuencia

   -- CUANTAS HORAS EXTRA -- OK --

   LET arr_horasdia[1] = vgrtabla.hrs_lun
   LET arr_horasdia[2] = vgrtabla.hrs_mar
   LET arr_horasdia[3] = vgrtabla.hrs_mie
   LET arr_horasdia[4] = vgrtabla.hrs_jue
   LET arr_horasdia[5] = vgrtabla.hrs_vie
   LET arr_horasdia[6] = vgrtabla.hrs_sab
   LET arr_horasdia[7] = vgrtabla.hrs_dom

   LET nDiasTX = 0
   LET nHoras2 = 0
   LET nHoras3 = 0

   FOR nI = 1 TO 7
      IF arr_horasdia[nI] != 0 THEN
         LET nDiasTX = nDiasTX + 1
         IF nDiasTX > 3 THEN
            LET nHoras3 = nHoras3 + arr_horasdia[nI]
         ELSE
            IF arr_horasdia[nI] > 3 THEN
               LET nHoras2 = nHoras2 + 3
               LET nHoras3 = nHoras3 + arr_horasdia[nI] - 3
```

```
            ELSE
               LET nHoras2 = nHoras2 + arr_horasdia[nI]
            END IF
         END IF
      END IF
   END FOR


   DISPLAY nHoras2, nHoras3 TO nX2, nX3


   LET nHoras2 = nHoras2 - nAntHoras2
   LET nHoras3 = nHoras3 - nAntHoras3

   LET arr_tiempoextra[nIndice].textra = (2 * nSueldoHora * nHoras2) + (3 * nSueldoHora * nHoras3)


   -- DIAS DE DESCANSO -- OK --

   LET arr_tiempoextra[nIndice].descanso = ((vgrtabla.dias_descanso - vgrtabla.pag_descanso) * 2 *
(nSueldoDiario))



   -- EXENTOS Y GRAVADOS DE HORAS EXTRA -- OK --

   IF ((arr_tiempoextra[nIndice].textra + nAntTextra + arr_tiempoextra[nIndice].descanso + nAntDescanso) / 2) >
(5 * nSalMin) THEN
      IF nAntExento < (5 * nSalMin) THEN
         LET arr_tiempoextra[nIndice].exento = (5 * nSalMin) - nAntExento
      ELSE
         LET arr_tiempoextra[nIndice].exento = 0
      END IF
      LET arr_tiempoextra[nIndice].gravado = arr_tiempoextra[nIndice].textra + arr_tiempoextra[nIndice].descanso
- arr_tiempoextra[nIndice].exento
   ELSE
      LET arr_tiempoextra[nIndice].exento  = ((arr_tiempoextra[nIndice].textra + nAntTextra +
                                               arr_tiempoextra[nIndice].descanso + nAntDescanso) / 2) -
nAntExento
      LET arr_tiempoextra[nIndice].gravado = arr_tiempoextra[nIndice].textra + arr_tiempoextra[nIndice].descanso
- arr_tiempoextra[nIndice].exento
   END IF



   -- PRIMA DOMINICAL Y SUS EXENTOS Y GRAVADOS -- OK --

   IF vgrtabla.pag_prima_dom = 'N' THEN
      IF vgrtabla.prima_dom = 'S' THEN
         LET arr_tiempoextra[nIndice].prima_dom = nSueldoDiario / 4

         IF arr_tiempoextra[nIndice].prima_dom > nSalMin THEN
            LET arr_tiempoextra[nIndice].exento  = arr_tiempoextra[nIndice].exento  + nSalMin
            LET arr_tiempoextra[nIndice].gravado = arr_tiempoextra[nIndice].gravado +
arr_tiempoextra[nIndice].prima_dom - nSalMin
         ELSE
            LET arr_tiempoextra[nIndice].exento  = arr_tiempoextra[nIndice].exento  +
arr_tiempoextra[nIndice].prima_dom
         END IF

      ELSE

         IF arr_tiempoextra[nIndice].prima_dom > nSalMin THEN
            LET arr_tiempoextra[nIndice].exento  = arr_tiempoextra[nIndice].exento  - nSalMin
            LET arr_tiempoextra[nIndice].gravado = arr_tiempoextra[nIndice].gravado -
arr_tiempoextra[nIndice].prima_dom + nSalMin
         ELSE
            LET arr_tiempoextra[nIndice].exento  = arr_tiempoextra[nIndice].exento  -
arr_tiempoextra[nIndice].prima_dom
         END IF

         LET arr_tiempoextra[nIndice].prima_dom = 0

      END IF
   END IF


   DISPLAY arr_tiempoextra[nIndice].* TO scr_tiempoextra[nIndice].*

   CALL say_totales()

END FUNCTION
################################################################################
```

- **CÁLCULO DEL IMPUESTO:**

Se desarrolló un Procedimiento Almacenado (SPL) que sólo calcula el monto del impuesto (ISR), cuya cohesión es muy alta y con un bajo acoplamiento, requiriendo cinco argumentos.

Se optó por generar un Procedimiento Almacenado (SPL) en lugar de un programa (4gl), ya que al estar inserto en la Base de Datos, desde cualquier módulo ya sea otro SPL, programa 4gl, consulta de Microsoft Query o cualquier programa front-end puede ser llamado para calcular el impuesto y regresar solo un valor como cualquier función.

A continuación se muestra el código fuente del SPL del cálculo del impuesto.

```
CREATE PROCEDURE calc_ispt2

--###############################################################################
--##### Nombre SPL    : calc_ispt2.sql
--##### Descripcion   : Calculo del ISPT (segunda parte)
--##### Fecha         : Noviembre 8 de 2001
--##### Autor         : Armando Montiel
--##### Fecha         : Enero 28 de 2003 / Mzo 26 de 2007 / Oct 26 de 2007
--#####               : Nov 14 de 2007 / Abr 30 de 2011
--##### Base de Datos : bd3_nomi
--#####               : Nuevo procedimiento de calculo que incluye retroactivos
--###############################################################################

   (
   cTipoNomina          CHAR(2),
   cNumEmp              CHAR(6),
   nSueldoBase          DECIMAL(14,2),
   nDiasFalta           SMALLINT,
   nDiasRetro           SMALLINT
   )

   RETURNING DECIMAL(14,2);           -- Monto del ISPT


   DEFINE cCveTabla           CHAR(1);

   DEFINE nI                  SMALLINT;
   DEFINE nVeces              SMALLINT;

   DEFINE nIngRetro           DECIMAL(14,2);
   DEFINE nImporte            DECIMAL(14,2);
   DEFINE nImpMens            DECIMAL(14,2);
   DEFINE nMesFiscal          DECIMAL(4,2);
   DEFINE nISPTn              DECIMAL(14,2);
   DEFINE nISPT               DECIMAL(14,2);
   DEFINE nExedLimInf         DECIMAL(14,2);
   DEFINE nImptoMarg          DECIMAL(14,2);
   DEFINE nDiferencia         DECIMAL(14,2);
   DEFINE nCredSal            DECIMAL(14,2);

   DEFINE nLimInf_80          DECIMAL(14,2);
   DEFINE nCFISPT_80          DECIMAL(14,2);
   DEFINE nEXISPT_80          DECIMAL(7,6);
   DEFINE nImptoTot           DECIMAL(14,2);

   DEFINE nLimInf_80a         DECIMAL(14,2);
   DEFINE nCFSUBS_80a         DECIMAL(14,2);
   DEFINE nEXSUBS_80a         DECIMAL(7,6);
   DEFINE nSubsidTot          DECIMAL(14,2);
   DEFINE nSubAcred           DECIMAL(14,2);


-- SET DEBUG FILE TO "calc_ispt2.out";
-- TRACE ON;

   LET cCveTabla   = "";

   LET nMesFiscal  = 30.4;
   LET nISPT       = 0;
   LET nExedLimInf = 0;
   LET nImptoMarg  = 0;
   LET nDiferencia = 0;
   LET nCredSal    = 0;

   LET nLimInf_80  = 0;
   LET nCFISPT_80  = 0;
   LET nEXISPT_80  = 0;
   LET nImptoTot   = 0;

   LET nLimInf_80a = 0;
   LET nCFSUBS_80a = 0;
   LET nEXSUBS_80a = 0;
   LET nSubsidTot  = 0;
   LET nSubAcred   = 0;

   LET cCveTabla = "1";
   LET nVeces    = 1;

   -- SI SE TRATA DE RETROACTIVO

   IF nDiasRetro != 0 THEN

      IF nDiasRetro > 1000 THEN

         LET nDiasRetro = nDiasRetro - 1000;
```

```
            LET nVeces = 2;

            LET nIngRetro = nSueldoBase;
            LET nImpMens  = nSueldoBase / nDiasRetro * nMesFiscal;

            SELECT (d01_impcon + d01_recmen + 2 * (d04_depare + d04_despen + d04_presoc))
               INTO nSueldoBase
               FROM m301_clpg, m320_memp, d304_prgr
               WHERE c20_numemp = cNumEmp    AND
                     c20_tipnom = c01_tipnom AND
                     c20_cpsn   = c01_clapag AND
                     c04_tipnom = c01_tipnom;

            LET nImporte = nImpMens + nSueldoBase;

         ELSE

            LET nIngRetro   = nSueldoBase;
            LET nSueldoBase = nSueldoBase / nDiasRetro * nMesFiscal;

         END IF

      END IF

      FOR nI = 1 TO nVeces

         IF nI = 2 THEN
            LET nISPTn = nISPT;
            LET nSueldoBase = nImporte;
         END IF


         -- IMPUESTO

         SELECT d07_liminf, d07_cfispt, d07_exispt / 100
            INTO nLimInf_80, nCFISPT_80, nEXISPT_80
            FROM m307_ispt
            WHERE c07_cvetab = cCveTabla AND
                  nSueldoBase BETWEEN d07_liminf AND d07_limsup;

         LET nExedLimInf = nSueldoBase - nLimInf_80;
         LET nImptoMarg  = nExedLimInf * nEXISPT_80;
         LET nImptoTot   = nImptoMarg  + nCFISPT_80;


         -- SUBSIDIO

         LET nDiferencia = nImptoTot - nSubAcred;


         -- SUBSIDIO PARA EL EMPLEO (ANTES CREDITO AL SALARIO)

         IF cTipoNomina = "11" OR cTipoNomina = "40" OR cTipoNomina = "50" THEN
            SELECT importe
               INTO nCredSal
               FROM crealsal
               WHERE cvetab = "1" AND
                     nSueldoBase BETWEEN para AND hasta;

            LET nCredSal = (nCredSal / 30) * (30 - nDiasFalta);

         ELSE
            LET nCredSal = 0;
         END IF

         IF nDiferencia < nCredSal THEN
            LET nISPT = nDiferencia;
         ELSE
            LET nISPT = nDiferencia - nCredSal;
         END IF


      END FOR


      -- RETROACTIVO POR REGLAMENTO

      IF nDiasRetro != 0 THEN
         IF nVeces = 2 THEN
            LET nISPT = nIngRetro * (nISPT - nISPTn) / nImpMens;
         ELSE
            LET nISPT = nISPT / nSueldoBase * nIngRetro;
         END IF
      END IF

      IF nISPT IS NULL THEN
         LET nISPT = 0;
      END IF

      RETURN nISPT;

   END PROCEDURE;
   --#############################################################################
```

12 ELABORACIÓN DE PROTOTIPOS

Uno de los nuevos procesos a desarrollar es el de pagos a terceros. El primero con el que se empezó a trabajar fue el Pago de Cuotas Sindicales.

Derivado de las actividades de la **Ingeniería de Requisitos**, el proceso debe generar las CXP de los tres diferentes sindicatos a partir de los descuentos que se les hace a los empleados de base y dependiendo a qué sindicato estén afiliados. Este proceso evita que tengan que ser capturadas las CXP en el área de Presupuesto.

Como primer prototipo, se le presentó al usuario la Pantalla de Consulta de Cuotas Sindicales, mostrada en la figura 12.1, en donde se efectuarían las consultas de las cuotas sindicales por quincena, para posteriormente ser generadas las CXP correspondientes; y se le explicó el funcionamiento que tendría dicho módulo. A partir de ese momento al usuario le pareció adecuado el formato de la pantalla y la funcionalidad que tendría, así que se continuó desarrollando en un par de iteraciones más del modelo de **Reingeniería de Procesos de Negocios** (RPN).

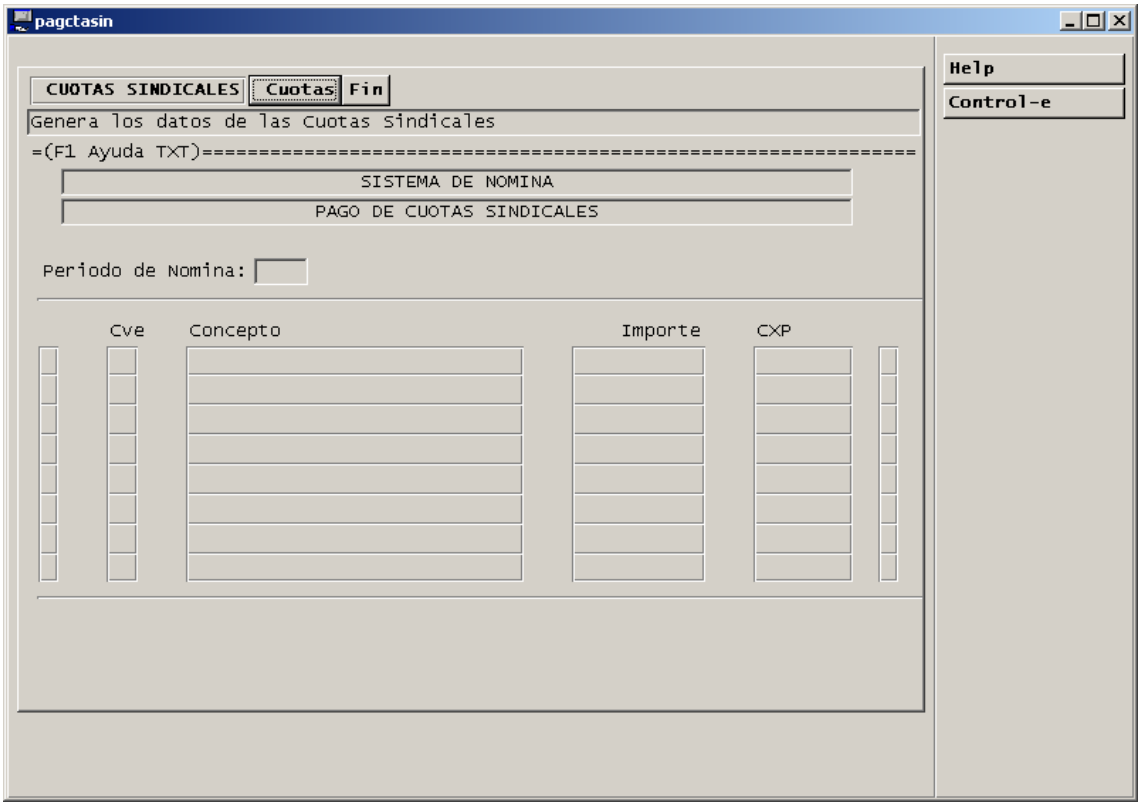


Figura 12.1 Pantalla de Consulta de Cuotas Sindicales

13 REFINAMIENTO E INSTANCIACIÓN (PERSONALIZACIÓN)

Continuando con el modelo de **Reingeniería de Procesos de Negocios** en la actividad de **Refinamiento** e **Instanciación** para el proceso de Pago de Cuotas Sindicales se agregaron las siguientes funcionalidades de acuerdo a la **Especificación** del usuario:

- **Cuotas** (consulta). Los descuentos deben de tomarse de la nómina correspondiente al periodo que se desee pagar, sumando los conceptos 63 (cuota sindical) y 64 (retroactivo de cuota sindical) de cada empleado, agrupándolos por tipo de concepto y sindicato. En el caso de que el empleado esté aportando al FONAC y esté afiliado al sindicato 1 (STALDF), solo se le sumará el 75% del concepto a dicho sindicato.
- **Detalle.** Debe de pasarse el control a los renglones que aparecen abajo por si existen más renglones de los desplegados en la pantalla.
- **Genera CXP.** Las CXP deben generarse después de consultar un periodo de nómina, presionando el botón "Generar CXP" y cuando ya estén generadas el botón deberá de desaparecer, de igual manera cuando se consulte un periodo de nómina en el que ya se hayan generado las CXP correspondientes y mostrando las CXP que les haya tocado a cada sindicato.
- **Reporte.** Generar un reporte en papel de lo consultado y mostrado en la pantalla.

Para tener un control de las CXP generadas para cada uno de los Pagos a Terceros y verificar que solo se hayan generado una sola vez, se diseñó una tabla de control llamada `sn_terceros`, mostrada en la figura 13.1. Si al efectuar la consulta de los conceptos de la nómina de un periodo y éste no existe en la tabla de control se mostrará el botón "`Genera CXP`" y se podrán generar las CXP correspondientes, de lo contrario el botón "`Genera CXP`" no será visible, se desplegará el mensaje "`Ya fueron Generadas las CXP para este periodo y Nomina`" y las CXP para cada concepto y Sindicato, en cuyo caso solo serviría de consulta. A continuación se muestra la tabla correspondiente después de analizar todos los Pagos a Terceros que son requeridos.

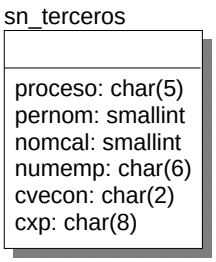


Figura 13.1 Tabla para el control de las CXP de terceros

En esta tabla se guardará una clave asignada para cada tipo de proceso (`proceso`), el periodo de la nómina (`pernom`), la clave de cálculo (`nomcal`), número de empleado (`numemp`), clave de concepto (`cvecon`) y cuenta por pagar (`cxp`). No todas las columnas son utilizadas y dependen del tipo de proceso; las únicas columnas requeridas son: tipo de proceso, periodo de la nómina y cuenta por pagar. La columna número de empleado se utiliza tanto para el número del empleado en las pensiones alimenticias como para indicar el sindicato al que se le está haciendo el pago.

En la primera iteración de Refinamiento se hizo la consulta a las tablas de la nómina sumando los importes de los conceptos 63 y 64 de todos los empleados de base y clasificándolos en los diferentes sindicatos, y de esta forma presentarlos en la pantalla como se muestra en la figura 13.2 Consulta de Cuotas Sindicales sin CXP generadas.

Una vez hecha la **Validación** por el usuario, en la segunda iteración se generó la CXP correspondiente, viéndose como se muestra en la figura 13.3 Consulta de Cuotas Sindicales con CXP generadas.

Parte del código fuente para seleccionar los datos y mostrarlos en la pantalla se muestra a continuación, así como las funciones de las opciones `Detalle` y `Genera CXP`, asimismo se muestra el reporte de una de las CXP generadas en la figura 13.4.

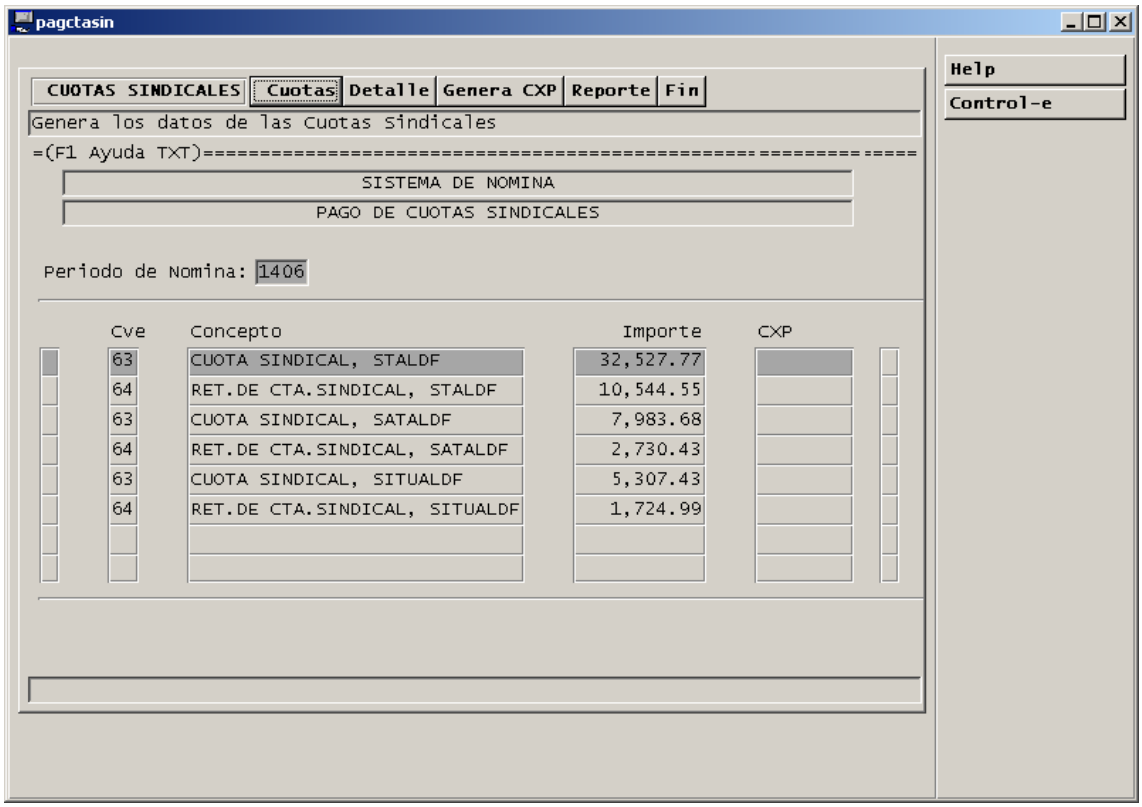


Figura 13.2 Consulta de Cuotas Sindicales sin CXP generadas

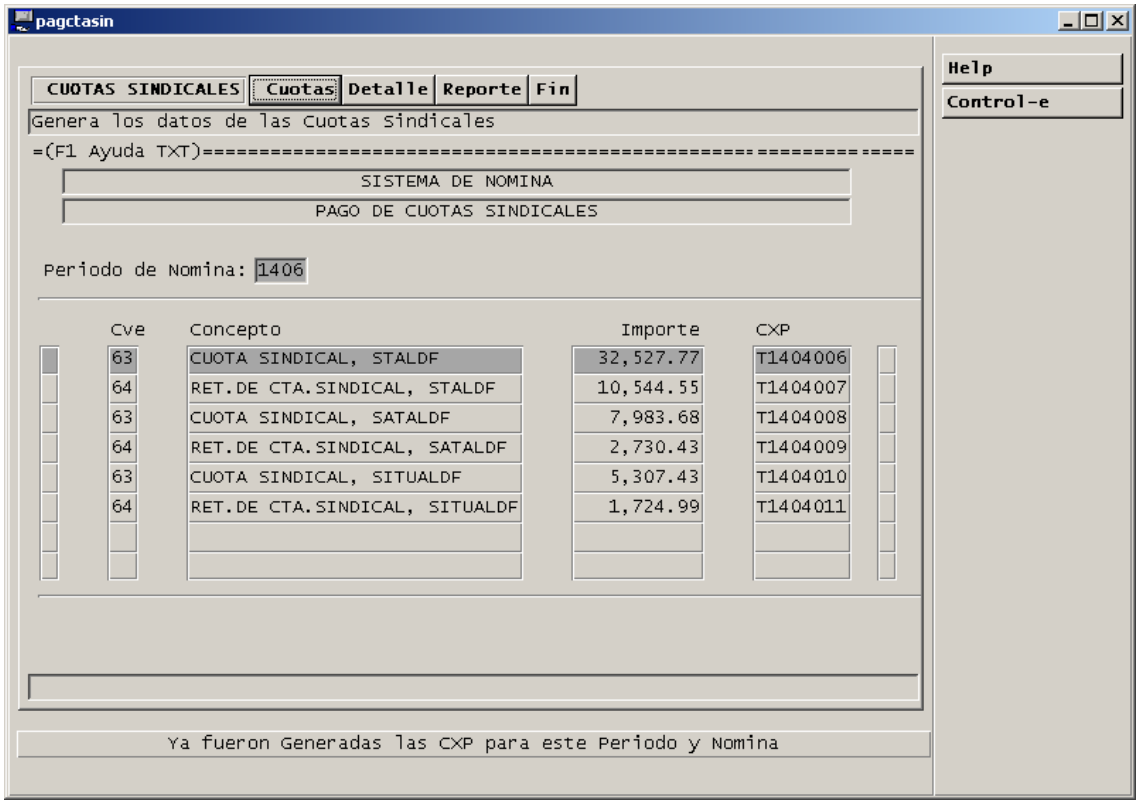


Figura 13.3 Consulta de Cuotas Sindicales con CXP generadas

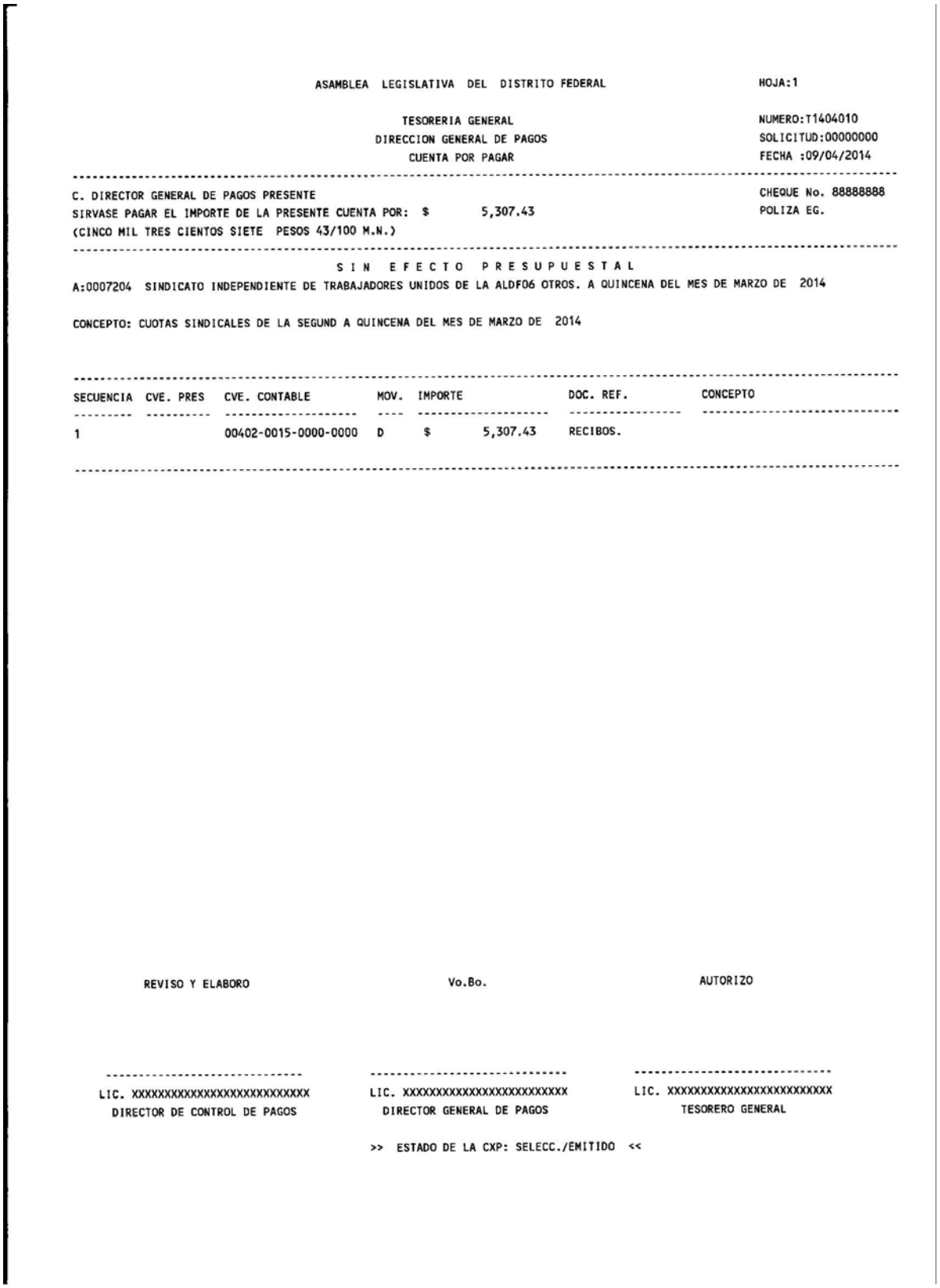
Funciones de las opciones `Cuotas, Detalle` y `Genera CXP`

```
##############################################################################
FUNCTION busca_ctasin()
   DEFINE
      cCveBene          CHAR(7),
      cNomBene          CHAR(30),
      nCont             SMALLINT,
      r_apopar          RECORD
         cvecon            CHAR(2),
         sindicato         SMALLINT,
         numemp            CHAR(6),
         importe           DECIMAL(12,2)
         END RECORD,
      r_ctasind         RECORD
         cvecon            CHAR(2),
         sindicato         SMALLINT,
         importe           DECIMAL(12,2)
         END RECORD,
      nCtaSind          DECIMAL(12,2)


   FOR vg_cont4 = 1 TO 20
      INITIALIZE arr_pagapo[vg_cont4].* TO NULL
   END FOR

   LET vg_cont2 = 0
   LET vg_cont3 = 0

   LET nImporteTot = 0

   CREATE TEMP TABLE cuotasind
      (
      cvecon            CHAR(2),
      sindicato         SMALLINT,
      importe           DECIMAL(12,2)
      )


   DECLARE c_desglo1 CURSOR FOR
      SELECT c12_cvecon, d12_salant, c12_numemp, SUM(d12_impor)
         FROM d312_mred
         WHERE c12_tipnom = '11' AND s12_pernom = nPeriodo AND c12_cvecon IN ('63','64')
         GROUP BY c12_cvecon, d12_salant, c12_numemp
         ORDER BY c12_cvecon, d12_salant, c12_numemp

   FOREACH c_desglo1
      INTO r_apopar.*

      LET nCont = 0

      LET nCtaSind = r_apopar.importe

      SELECT COUNT(*)
         INTO nCont
         FROM d311_mrep
         WHERE c11_tipnom = '11'            AND
               s11_pernom = nPeriodo        AND
               c11_numemp = r_apopar.numemp AND
               c11_cvecon = '41'

      IF nCont IS NOT NULL AND nCont != 0 AND r_apopar.sindicato = 1 THEN
         LET r_apopar.importe = nCtaSind - (nCtaSind * 0.25)
      END IF


      DISPLAY r_apopar.numemp,' ',r_apopar.sindicato,' ',r_apopar.cvecon,' ',nCtaSind,'
',r_apopar.importe

      SELECT *
         INTO r_ctasind.*
         FROM cuotasind
         WHERE cvecon    = r_apopar.cvecon AND
               sindicato = r_apopar.sindicato

      IF STATUS = NOTFOUND THEN
         INSERT INTO cuotasind
            VALUES (r_apopar.cvecon, r_apopar.sindicato, r_apopar.importe)
      ELSE
         UPDATE cuotasind
            SET importe = importe + r_apopar.importe
            WHERE cvecon    = r_apopar.cvecon AND
                  sindicato = r_apopar.sindicato
      END IF

   END FOREACH

   LET _Usuario = FGL_GETENV("LOGNAME")
   LET _Fecha   = TODAY
   LET _Hora    = CURRENT HOUR TO SECOND


   DECLARE c_desglo2 CURSOR FOR
      SELECT *
         FROM cuotasind
         WHERE importe != 0
         ORDER BY 2,1
```

```
   FOREACH c_desglo2
      INTO r_ctasind.*

      LET vg_cont3 = vg_cont3 + 1
      LET arr_pagapo[vg_cont3].x       = ' '
      LET arr_pagapo[vg_cont3].cvecon  = r_ctasind.cvecon
      LET arr_pagapo[vg_cont3].importe = r_ctasind.importe
      LET arr_pagapo[vg_cont3].y       = r_ctasind.sindicato

      SELECT c03_deslar
         INTO arr_pagapo[vg_cont3].descrip
         FROM d303_mpyd
         WHERE c03_tipnom = '11' AND
               c03_cvecon = r_ctasind.cvecon

      CASE
         WHEN r_ctasind.sindicato = 1
            LET arr_pagapo[vg_cont3].descrip = arr_pagapo[vg_cont3].descrip CLIPPED, ', STALDF'

         WHEN r_ctasind.sindicato = 2
            LET arr_pagapo[vg_cont3].descrip = arr_pagapo[vg_cont3].descrip CLIPPED, ', SATALDF'

         WHEN r_ctasind.sindicato = 3
            LET arr_pagapo[vg_cont3].descrip = arr_pagapo[vg_cont3].descrip CLIPPED, ', SITUALDF'

      END CASE

      LET arr_sindicato[vg_cont3] = r_ctasind.sindicato

      SELECT cxp
         INTO arr_pagapo[vg_cont3].cxp
         FROM sn_terceros
         WHERE proceso = cTipoProc          AND
               pernom  = nPeriodo           AND
               numemp  = r_ctasind.sindicato AND
               cvecon  = r_ctasind.cvecon


      LET _Referencia = 'PERIODO: ',nPeriodo USING '&&&&',', SINDICATO: ',r_ctasind.sindicato,', CXP: ',
                        arr_pagapo[vg_cont3].cxp,', IMPORTE: ',arr_pagapo[vg_cont3].importe

      INSERT INTO db_log:bitacora
         VALUES (0, cSistema, _Usuario, _Fecha, _Hora, 'PPAGCTASIN', 'C', _Referencia)

   END FOREACH

   DROP TABLE cuotasind

   LET cCXP = ''

   IF arr_pagapo[1].cxp IS NOT NULL THEN
      LET vg_cont2 = vg_cont3
   END IF

   CALL SET_COUNT(vg_cont3)

   DISPLAY arr_pagapo[8].* TO scr_pagapo[8].*
   DISPLAY arr_pagapo[7].* TO scr_pagapo[7].*
   DISPLAY arr_pagapo[6].* TO scr_pagapo[6].*
   DISPLAY arr_pagapo[5].* TO scr_pagapo[5].*
   DISPLAY arr_pagapo[4].* TO scr_pagapo[4].*
   DISPLAY arr_pagapo[3].* TO scr_pagapo[3].*
   DISPLAY arr_pagapo[2].* TO scr_pagapo[2].*
   DISPLAY arr_pagapo[1].* TO scr_pagapo[1].*

   RETURN vg_cont3

END FUNCTION
################################################################################
FUNCTION detalle_ctasin()
   DEFINE
      nKey,
      nScrRow,
      nRegTot,
      nRegRow           SMALLINT


   INPUT ARRAY arr_pagapo
      WITHOUT DEFAULTS
      FROM scr_pagapo.*

      BEFORE ROW
         LET nRegRow = ARR_CURR()
         LET nScrRow = SCR_LINE()
         LET arr_pagapo[nRegRow].x = ">"
         DISPLAY arr_pagapo[nRegRow].x TO scr_pagapo[nScrRow].x

      AFTER FIELD x
         LET nRegRow = ARR_CURR()
         LET nScrRow = SCR_LINE()
         LET arr_pagapo[nRegRow].x = " "
         DISPLAY arr_pagapo[nRegRow].x TO scr_pagapo[nScrRow].x

         LET nKey = FGL_LASTKEY()
         IF nKey = FGL_KEYVAL("DOWN") OR nKey = ENTER OR nKey = TAB OR nKey = RIGHT THEN
```

```
            IF ARR_CURR() >= vg_cont3 THEN
               LET arr_pagapo[nRegRow].x = ">"
               DISPLAY arr_pagapo[nRegRow].x TO scr_pagapo[nScrRow].x
               ERROR " There are no more rows in the direction you are going "
               NEXT FIELD x
            END IF
         END IF

      ON KEY ("CONTROL-E")
         CALL calendar()

   END INPUT

   IF INT_FLAG THEN
      LET INT_FLAG = FALSE
   END IF

END FUNCTION
################################################################################
FUNCTION cxp_ctasin()
   DEFINE
      vban              CHAR(1),
      cRefere           CHAR(45),
      cPeriodo          CHAR(4),
      cAnio             CHAR(4),
      cMes              CHAR(2),
      cCXP              CHAR(8),
      cSubCta           CHAR(4),
      nHay,
      i,j,k             SMALLINT,
      cBeneficiario     CHAR(30),
      r_m109_vums       RECORD LIKE bd1_sitg:m109_vums.*,
      cMensaje          CHAR(120)


   SELECT f06_fecfin
      INTO dFecha
      FROM c306_prno
      WHERE c06_tipnom = cNomina AND
            s06_nomcal = 1       AND
            s06_pernom = nPeriodo


   -- AJUSTE PARA GENERAR LAS CUENTAS POR PAGAR DE LA SEGUNDA QUINCENA CON FECHA DEL SIGUENTE MES
(PERIODO)

   LET cPeriodo = nPeriodo USING '&&&&'
   LET i        = cPeriodo[3,4]

   IF (i MOD 2) = 1 OR i = 24 THEN
      LET cMes = MONTH(dFecha) USING '&&'
   ELSE
      LET cMes = (MONTH(dFecha) + 1) USING '&&'
   END IF


   LET cCXP  = 'T', cPeriodo[1,2], cMes, '*'
   LET cAnio = '20', cPeriodo[1,2]


   -- ASIGNA LOS VALORES FIJOS DE LA CXP.

   IF i = 24 THEN
      LET r_m109_vums.f09_fecreg = dia_fin_mes(dFecha,'')
   ELSE
      LET r_m109_vums.f09_fecreg = TODAY
   END IF

   LET r_m109_vums.c09_proyec = '02'
   LET r_m109_vums.c09_anuafe = cAnio
   LET r_m109_vums.c09_mesafe = cMes
   LET r_m109_vums.f09_fecpag = ''
   LET r_m109_vums.c09_tipope = '1'
   LET r_m109_vums.c09_edovum = '02'
   LET r_m109_vums.d09_sdovum = 0
   LET r_m109_vums.c09_firmae = '36'      -- director de control de pagos
   LET r_m109_vums.c09_firmar = '30'      -- director general de pagos
   LET r_m109_vums.c09_firmaa = '29'      -- tesorero
   LET r_m109_vums.c09_tidesc = '06'
   LET r_m109_vums.c09_despol = 'OTROS.'
   LET r_m109_vums.c09_edoafe = 'A'
   LET r_m109_vums.c09_origen = '4'
   LET r_m109_vums.c09_numche = ''

   SELECT COUNT(*)
      INTO nHay
      FROM bd1_sitg:m109_vums
      WHERE c09_numvum MATCHES cCXP

   IF nHay IS NULL THEN
      LET nHay = 0
   END IF

   LET _Usuario = FGL_GETENV("LOGNAME")
   LET _Fecha   = TODAY
   LET _Hora    = CURRENT HOUR TO SECOND
```

```
   BEGIN WORK

   FOR i = 1 TO vg_cont3

      LET j = nHay + i

      LET r_m109_vums.c09_numvum = cCXP[1,5], j USING '&&&'
      LET r_m109_vums.d09_impvum = arr_pagapo[i].importe
      LET arr_pagapo[i].cxp      = r_m109_vums.c09_numvum

      IF i = 1 THEN
         LET cMsgPolizas = ' Se generaron las cuentas por pagar de la ',r_m109_vums.c09_numvum
      END IF

      LET cMensaje = 'CUOTAS SINDICALES'


      IF arr_pagapo[i].cvecon = '64' THEN
         LET cMensaje = 'RETROACTIVO DE ', cMensaje
      END IF

      CASE
         WHEN arr_sindicato[i] = 1
            LET r_m109_vums.c09_cvepro = '0001340'
            LET r_m109_vums.c09_tiprov = '3'
            LET cSubCta                = '0012'
         WHEN arr_sindicato[i] = 2
            LET r_m109_vums.c09_cvepro = '0005657'
            LET r_m109_vums.c09_tiprov = '4'
            LET cSubCta                = '5657'
         WHEN arr_sindicato[i] = 3
            LET r_m109_vums.c09_cvepro = '0007204'
            LET r_m109_vums.c09_tiprov = '4'
            LET cSubCta                = '0015'
      END CASE

      LET k = MONTH(dFecha)

      IF DAY(dFecha) = 15 THEN
         LET cMensaje = cMensaje CLIPPED, ' DE LA PRIMERA QUINCENA DEL MES DE ',arr_meses[k] CLIPPED,
                        ' DE ',YEAR(dFecha) USING '&&&&'
      ELSE
         LET cMensaje = cMensaje CLIPPED, ' DE LA SEGUNDA QUINCENA DEL MES DE ',arr_meses[k] CLIPPED,
                        ' DE ',YEAR(dFecha) USING '&&&&'
      END IF

      LET r_m109_vums.c09_conce1 = cMensaje[  1, 30]
      LET r_m109_vums.c09_conce2 = cMensaje[ 31, 60]
      LET r_m109_vums.c09_conce3 = cMensaje[ 61, 90]
      LET r_m109_vums.c09_conce4 = cMensaje[ 91,120]

      SELECT c18_nombre
         INTO cBeneficiario
         FROM bd1_sitg:m118_bene
         WHERE c18_cvepro = r_m109_vums.c09_cvepro AND
               c18_tiprov = r_m109_vums.c09_tiprov


      INSERT INTO bd1_sitg:m109_vums
         VALUES (r_m109_vums.*)

      LET cRefere = '000',r_m109_vums.c09_cvepro

      INSERT INTO bd1_sitg:d110_parv09
         VALUES (r_m109_vums.c09_anuafe, r_m109_vums.c09_mesafe, r_m109_vums.c09_numvum, 1, 'C', '0000',
                 '00402', cSubCta, '0000', '0000', 'D', cRefere, r_m109_vums.d09_impvum, '02', 'C')

      INSERT INTO bd1_sitg:m150_emis
         VALUES ('00000000', r_m109_vums.c09_cvepro, r_m109_vums.c09_tiprov, '00000000', cBeneficiario,
                 r_m109_vums.f09_fecreg, TODAY, '00', r_m109_vums.d09_impvum, r_m109_vums.c09_conce1,
                 r_m109_vums.c09_conce2, r_m109_vums.c09_conce3, r_m109_vums.c09_conce4,
                 r_m109_vums.c09_proyec, r_m109_vums.c09_numvum, 'G', '1', '4', '0000000000',
'00000000')

      INSERT INTO sn_terceros
         VALUES (cTipoProc, nPeriodo, '', arr_sindicato[i], arr_pagapo[i].cvecon,
r_m109_vums.c09_numvum)


      LET _Referencia = 'PERIODO: ',nPeriodo USING '&&&&',', SINDICATO: ',arr_sindicato[i],', CXP:
',arr_pagapo[i].cxp,', IMPORTE: ',arr_pagapo[i].importe

      INSERT INTO db_log:bitacora
         VALUES (0, cSistema, _Usuario, _Fecha, _Hora, 'PPAGCTASIN', 'P', _Referencia)

   END FOR

   LET cMsgPolizas = cMsgPolizas CLIPPED, ' a la ',r_m109_vums.c09_numvum

   COMMIT WORK

END FUNCTION
################################################################################
```



ASAMBLEA LEGISLATIVA DEL DISTRITO FEDERAL — HOJA:1

TESORERIA GENERAL
DIRECCION GENERAL DE PAGOS
CUENTA POR PAGAR

NUMERO:T1404010
SOLICITUD:00000000
FECHA :09/04/2014

C. DIRECTOR GENERAL DE PAGOS PRESENTE
SIRVASE PAGAR EL IMPORTE DE LA PRESENTE CUENTA POR: $ 5,307.43
(CINCO MIL TRES CIENTOS SIETE PESOS 43/100 M.N.)

CHEQUE No. 88888888
POLIZA EG.

S I N E F E C T O P R E S U P U E S T A L

A:0007204 SINDICATO INDEPENDIENTE DE TRABAJADORES UNIDOS DE LA ALDF06 OTROS. A QUINCENA DEL MES DE MARZO DE 2014

CONCEPTO: CUOTAS SINDICALES DE LA SEGUND A QUINCENA DEL MES DE MARZO DE 2014

SECUENCIA	CVE. PRES	CVE. CONTABLE	MOV.	IMPORTE	DOC. REF.	CONCEPTO
1		00402-0015-0000-0000	D	$ 5,307.43	RECIBOS.	

REVISO Y ELABORO

LIC. XXXXXXXXXXXXXXXXXXXXXXXXXXX
DIRECTOR DE CONTROL DE PAGOS

Vo.Bo.

LIC. XXXXXXXXXXXXXXXXXXXXXXXXXX
DIRECTOR GENERAL DE PAGOS

AUTORIZO

LIC. XXXXXXXXXXXXXXXXXXXXXXXXX
TESORERO GENERAL

>> ESTADO DE LA CXP: SELECC./EMITIDO <<

Figura 13.4 Reporte de una CXP generada

14 DESARROLLO RÁPIDO DE APLICACIONES (DRA)

Debido a que el enfoque de un DRA es la construcción de nuevos componentes basados en componentes de software existente, el desarrollo de los nuevos procesos de Pago a Terceros se hizo tomando como base para tal efecto el Pago de Cuotas Sindicales que se vio en los dos capítulos anteriores, dado que su aplicación es muy similar en todos los procesos de Pagos a Terceros, pues prácticamente hay que cambiar los conceptos de los descuentos de la nómina correspondiente para cada pago, las descripciones, las fechas de aplicación y si se genera una o varias CXP para el pago.

Dos ejemplos de estos procesos se muestran en las figuras 14.1 y 14.2:

- **Pago de Seguro de Metlife**

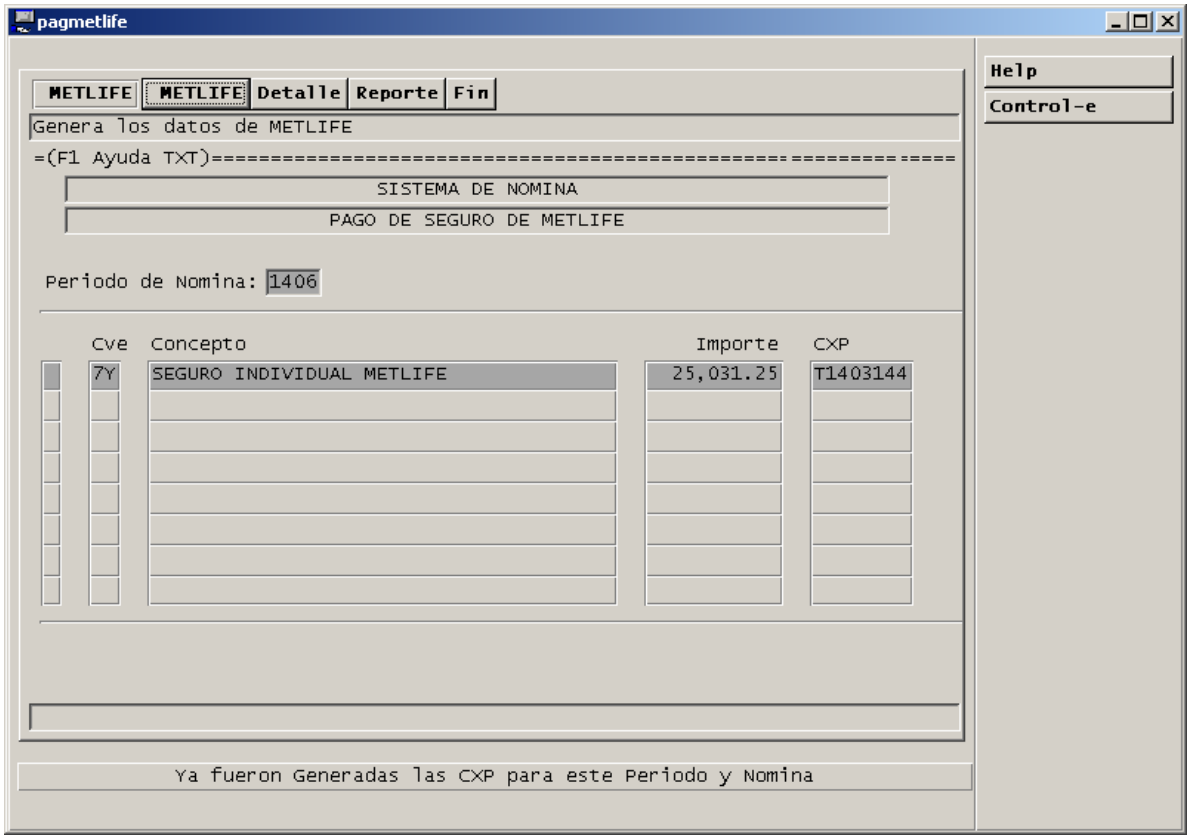


Figura 14.1 Consulta de Seguro de Metlife con CXP generada

Funciones de las opciones `METLIFE`, `Detalle` y `Genera CXP`:

```
##############################################################################
FUNCTION busca_aporta()
   DEFINE
      cCveBene          CHAR(7),
      cNomBene          CHAR(30),
      r_apopar          RECORD
         cvecon            CHAR(2),
         importe           DECIMAL(12,2)
         END RECORD


   FOR vg_cont4 = 1 TO 20
      INITIALIZE arr_pagapo[vg_cont4].* TO NULL
   END FOR
```

```
      LET vg_cont2 = 0
      LET vg_cont3 = 0

      LET nImporteTot = 0


      LET _Usuario = FGL_GETENV("LOGNAME")
      LET _Fecha   = TODAY
      LET _Hora    = CURRENT HOUR TO SECOND


      DECLARE c_desglo CURSOR FOR
         SELECT c12_cvecon, SUM(d12_impor)
            FROM d312_mred
            WHERE s12_pernom = nPeriodo AND c12_cvecon IN ('7Y')
            GROUP BY 1
            ORDER BY 1

      FOREACH c_desglo
         INTO r_apopar.*

         LET vg_cont3 = vg_cont3 + 1
         LET arr_pagapo[vg_cont3].x       = ' '
         LET arr_pagapo[vg_cont3].cvecon  = r_apopar.cvecon
         LET arr_pagapo[vg_cont3].importe = r_apopar.importe

         SELECT c03_deslar
            INTO arr_pagapo[vg_cont3].descrip
            FROM d303_mpyd
            WHERE c03_tipnom = '50' AND
                  c03_cvecon = r_apopar.cvecon

         SELECT cxp
            INTO arr_pagapo[vg_cont3].cxp
            FROM sn_terceros
            WHERE proceso = cTipoProc          AND
                  pernom  = nPeriodo           AND
                  cvecon  = r_apopar.cvecon


         LET _Referencia = 'PERIODO: ',nPeriodo USING '&&&&',', CONCEPTO: ',arr_pagapo[vg_cont3].cvecon,
                           ', CXP: ',arr_pagapo[vg_cont3].cxp,', IMPORTE: ',arr_pagapo[vg_cont3].importe

         INSERT INTO db_log:bitacora
            VALUES (0, cSistema, _Usuario, _Fecha, _Hora, 'PPAGMETLIFE', 'C', _Referencia)

      END FOREACH

      LET cCXP = ''

      IF arr_pagapo[1].cxp IS NOT NULL THEN
         LET vg_cont2 = vg_cont3
      END IF

      CALL SET_COUNT(vg_cont3)

      DISPLAY arr_pagapo[8].* TO scr_pagapo[8].* --ATTRIBUTE(REVERSE)
      DISPLAY arr_pagapo[7].* TO scr_pagapo[7].* --ATTRIBUTE(REVERSE)
      DISPLAY arr_pagapo[6].* TO scr_pagapo[6].* --ATTRIBUTE(REVERSE)
      DISPLAY arr_pagapo[5].* TO scr_pagapo[5].* --ATTRIBUTE(REVERSE)
      DISPLAY arr_pagapo[4].* TO scr_pagapo[4].* --ATTRIBUTE(REVERSE)
      DISPLAY arr_pagapo[3].* TO scr_pagapo[3].* --ATTRIBUTE(REVERSE)
      DISPLAY arr_pagapo[2].* TO scr_pagapo[2].* --ATTRIBUTE(REVERSE)
      DISPLAY arr_pagapo[1].* TO scr_pagapo[1].* --ATTRIBUTE(REVERSE)

      RETURN vg_cont3

   END FUNCTION
   ################################################################################
   FUNCTION detalle_aporta()
      DEFINE
         nKey,
         nScrRow,
         nRegTot,
         nRegRow           SMALLINT


      INPUT ARRAY arr_pagapo
         WITHOUT DEFAULTS
         FROM scr_pagapo.*

         BEFORE ROW
            LET nRegRow = ARR_CURR()
            LET nScrRow = SCR_LINE()
            LET arr_pagapo[nRegRow].x = ">"
            DISPLAY arr_pagapo[nRegRow].x TO scr_pagapo[nScrRow].x

         AFTER FIELD x
            LET nRegRow = ARR_CURR()
            LET nScrRow = SCR_LINE()
            LET arr_pagapo[nRegRow].x = " "
            DISPLAY arr_pagapo[nRegRow].x TO scr_pagapo[nScrRow].x
```

```
         LET nKey = FGL_LASTKEY()
         IF nKey = FGL_KEYVAL("DOWN") OR nKey = ENTER OR nKey = TAB OR nKey = RIGHT THEN
            IF ARR_CURR() >= vg_cont3 THEN
               LET arr_pagapo[nRegRow].x = ">"
               DISPLAY arr_pagapo[nRegRow].x TO scr_pagapo[nScrRow].x
               ERROR " There are no more rows in the direction you are going "
               NEXT FIELD x
            END IF
         END IF

      ON KEY ("CONTROL-E")
         CALL calendar()

   END INPUT

   IF INT_FLAG THEN
      LET INT_FLAG = FALSE
   END IF

END FUNCTION
################################################################################
FUNCTION cxp_aporta()
   DEFINE
      vban              CHAR(1),
      cRefere           CHAR(45),
      cPeriodo          CHAR(4),
      cAnio             CHAR(4),
      cMes              CHAR(2),
      cCXP              CHAR(8),
      cSubCta           CHAR(4),
      nHay,
      i,j,k             SMALLINT,
      cBeneficiario     CHAR(30),
      r_m109_vums       RECORD LIKE bd1_sitg:m109_vums.*,
      cMensaje          CHAR(120)


   SELECT f06_fecfin
      INTO dFecha
      FROM c306_prno
      WHERE c06_tipnom = '50' AND
            s06_nomcal = 1    AND
            s06_pernom = nPeriodo

{
   -- AJUSTE PARA GENERAR LAS CUENTAS POR PAGAR DE LA SEGUNDA QUINCENA CON FECHA DEL SIGUENTE MES
(PERIODO)

   LET cPeriodo = nPeriodo USING '&&&&'
   LET i        = cPeriodo[3,4]

   IF (i MOD 2) = 1 OR i = 24 THEN
      LET cMes = MONTH(dFecha) USING '&&'
   ELSE
      LET cMes = (MONTH(dFecha) + 1) USING '&&'
   END IF

   LET cCXP  = 'T', cPeriodo[1,2], cMes, '*'
   LET cAnio = '20', cPeriodo[1,2]


   -- ASIGNA LOS VALORES FIJOS DE LA CXP.

   IF i = 24 THEN
      LET r_m109_vums.f09_fecreg = dia_fin_mes(dFecha,'')
   ELSE
      LET r_m109_vums.f09_fecreg = TODAY
   END IF
}

   -- GENERAR LAS CUENTAS POR PAGAR

   LET cPeriodo = nPeriodo USING '&&&&'
   LET i        = cPeriodo[3,4]
   LET cMes     = MONTH(dFecha) USING '&&'
   LET cCXP     = 'T', cPeriodo[1,2], cMes, '*'
   LET cAnio    = '20', cPeriodo[1,2]


   -- ASIGNA LOS VALORES FIJOS DE LA CXP.

   LET r_m109_vums.f09_fecreg = TODAY

   LET r_m109_vums.c09_proyec = '02'
   LET r_m109_vums.c09_anuafe = cAnio
   LET r_m109_vums.c09_mesafe = cMes
   LET r_m109_vums.f09_fecpag = ''
   LET r_m109_vums.c09_tipope = '1'
   LET r_m109_vums.c09_edovum = '02'
   LET r_m109_vums.c09_cvepro = '0003839'
   LET r_m109_vums.c09_tiprov = '3'
   LET r_m109_vums.d09_sdovum = 0
   LET r_m109_vums.c09_firmae = '36'      -- director de control de pagos
   LET r_m109_vums.c09_firmar = '30'      -- director general de pagos
   LET r_m109_vums.c09_firmaa = '29'      -- tesorero
```

```
   LET r_m109_vums.c09_tidesc = '06'
   LET r_m109_vums.c09_despol = 'OTROS.'
   LET r_m109_vums.c09_edoafe = 'A'
   LET r_m109_vums.c09_origen = '4'
   LET r_m109_vums.c09_numche = ''

   SELECT COUNT(*)
      INTO nHay
      FROM bd1_sitg:m109_vums
      WHERE c09_numvum MATCHES cCXP

   IF nHay IS NULL THEN
      LET nHay = 0
   END IF

   LET _Usuario = FGL_GETENV("LOGNAME")
   LET _Fecha   = TODAY
   LET _Hora    = CURRENT HOUR TO SECOND


   BEGIN WORK

   FOR i = 1 TO vg_cont3

      LET j = nHay + i

      LET r_m109_vums.c09_numvum = cCXP[1,5], j USING '&&&'
      LET r_m109_vums.d09_impvum = arr_pagapo[i].importe
      LET arr_pagapo[i].cxp      = r_m109_vums.c09_numvum

      IF i = 1 THEN
         LET cMsgPolizas = ' Se generaron las cuentas por pagar de la ',r_m109_vums.c09_numvum
      END IF

      LET cMensaje = 'SEGURO DE VIDA METLIFE'

      LET k = MONTH(dFecha)

      IF DAY(dFecha) = 15 THEN
         LET cMensaje = cMensaje CLIPPED, ' DE LA PRIMERA QUINCENA DEL MES DE ',arr_meses[k] CLIPPED,
                        ' DE ',YEAR(dFecha) USING '&&&&'
      ELSE
         LET cMensaje = cMensaje CLIPPED, ' DE LA SEGUNDA QUINCENA DEL MES DE ',arr_meses[k] CLIPPED,
                        ' DE ',YEAR(dFecha) USING '&&&&'
      END IF

      LET r_m109_vums.c09_conce1 = cMensaje[  1, 30]
      LET r_m109_vums.c09_conce2 = cMensaje[ 31, 60]
      LET r_m109_vums.c09_conce3 = cMensaje[ 61, 90]
      LET r_m109_vums.c09_conce4 = cMensaje[ 91,120]

      SELECT c18_nombre
         INTO cBeneficiario
         FROM bd1_sitg:m118_bene
         WHERE c18_cvepro = r_m109_vums.c09_cvepro AND
               c18_tiprov = r_m109_vums.c09_tiprov


      INSERT INTO bd1_sitg:m109_vums
         VALUES (r_m109_vums.*)

      LET cRefere = '000',r_m109_vums.c09_cvepro

      INSERT INTO bd1_sitg:d110_parv09
         VALUES (r_m109_vums.c09_anuafe, r_m109_vums.c09_mesafe, r_m109_vums.c09_numvum, 1, 'C', '0000',
                 '00402', '0035', '0000', '0000', 'D', cRefere, r_m109_vums.d09_impvum, '02', 'C')

      INSERT INTO bd1_sitg:m150_emis
         VALUES ('00000000', r_m109_vums.c09_cvepro, r_m109_vums.c09_tiprov, '00000000', cBeneficiario,
                   r_m109_vums.f09_fecreg, TODAY, '00', r_m109_vums.d09_impvum, r_m109_vums.c09_conce1,
                   r_m109_vums.c09_conce2, r_m109_vums.c09_conce3, r_m109_vums.c09_conce4,
                   r_m109_vums.c09_proyec, r_m109_vums.c09_numvum, 'G', '1', '4', '0000000000',
'00000000')

--    DISPLAY cTipoProc,' ',nPeriodo,' ',' ',' ',arr_pagapo[i].cvecon,' ',r_m109_vums.c09_numvum

      INSERT INTO sn_terceros
         VALUES (cTipoProc, nPeriodo, '', '', arr_pagapo[i].cvecon, r_m109_vums.c09_numvum)


      LET _Referencia = 'PERIODO: ',nPeriodo USING '&&&&',', CONCEPTO: ',arr_pagapo[i].cvecon,', CXP: ',
                        arr_pagapo[i].cxp,', IMPORTE: ',arr_pagapo[i].importe

      INSERT INTO db_log:bitacora
         VALUES (0, cSistema, _Usuario, _Fecha, _Hora, 'PPAGMETLIFE', 'P', _Referencia)

   END FOR

   LET cMsgPolizas = cMsgPolizas CLIPPED, ' a la ',r_m109_vums.c09_numvum

   COMMIT WORK
-- ROLLBACK WORK

END FUNCTION
```

##

- **Pago de Pensión Alimenticia**

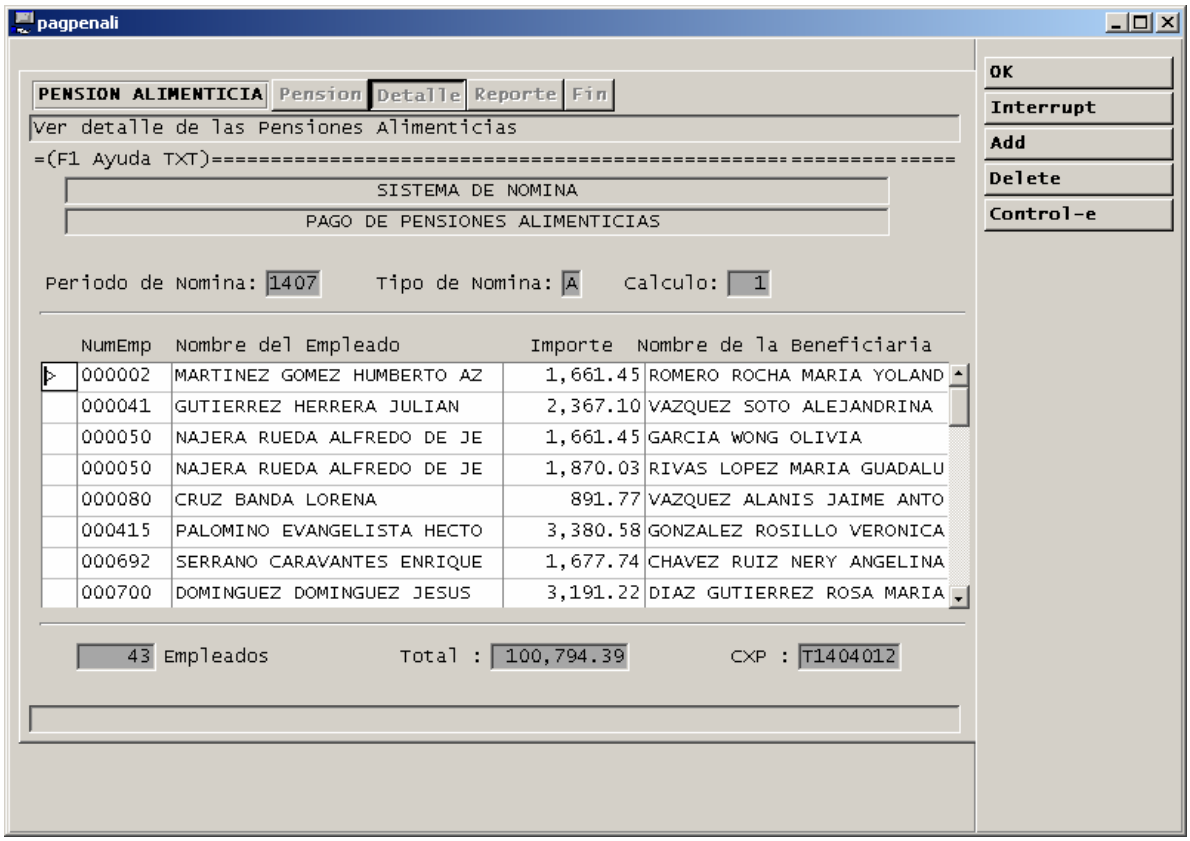


Figura 14.2 Consulta de Pensión Alimenticia con CXP generadas

Funciones de las opciones `Pension, Detalle` y `Genera CXP`:

```
################################################################################
FUNCTION busca_pen_alim()
   DEFINE
      cCveBene          CHAR(7),
      cNomBene          CHAR(30),
      r_sn_pen_alim     RECORD LIKE sn_pen_alim.*


   FOR vg_cont4 = 1 TO 200
      INITIALIZE arr_penali[vg_cont4].* TO NULL
      INITIALIZE arr_cuentas[vg_cont4].* TO NULL
   END FOR

   LET nErrores = 0
   LET vg_cont2 = 0
   LET vg_cont3 = 0

   LET nImporteTot = 0


   DECLARE c_desglo CURSOR FOR
      SELECT c12_tipnom, s12_nomcal, s12_pernom, c12_numemp, c12_cvecon, SUM(d12_impor)
         FROM d312_mred
         WHERE (cTipoNom = 'D' AND s12_pernom = nPeriodo AND s12_nomcal = nNomCal AND
                           c12_tipnom = '01' AND c12_cvecon IN ('77','87','59','7A')) OR
               (cTipoNom = 'A' AND s12_pernom = nPeriodo AND s12_nomcal = nNomCal AND
                           c12_tipnom IN ('11','40','50') AND c12_cvecon IN ('56','59','74','78')) OR
               (cTipoNom = 'A' AND s12_pernom = nPeriodo AND s12_nomcal = nNomCal AND
                           c12_tipnom IN ('50') AND c12_cvecon IN ('53','58')) OR
               (cTipoNom = 'A' AND s12_pernom = nPeriodo AND s12_nomcal = nNomCal AND
                           c12_tipnom IN ('20','21','25','35','70') AND c12_cvecon
                                                                 IN ('74','83','56','59'))
         GROUP BY 1,2,3,4,5
         ORDER BY 2,1,4
```

```
   FOREACH c_desglo
      INTO r_d312_mred.c12_tipnom, r_d312_mred.s12_nomcal, r_d312_mred.s12_pernom,
r_d312_mred.c12_numemp,
           r_d312_mred.c12_cvecon, r_d312_mred.d12_impor

      IF r_d312_mred.d12_impor != 0 THEN

         LET vg_cont3 = vg_cont3 + 1
         LET arr_penali[vg_cont3].x       = ' '
         LET arr_penali[vg_cont3].numemp  = r_d312_mred.c12_numemp
         LET arr_penali[vg_cont3].nomemp  = nombre_empleado(r_d312_mred.c12_numemp)
         LET arr_penali[vg_cont3].importe = r_d312_mred.d12_impor

         LET arr_cuentas[vg_cont3].nomcal = r_d312_mred.s12_nomcal
         LET arr_cuentas[vg_cont3].pernom = r_d312_mred.s12_pernom
         LET arr_cuentas[vg_cont3].cvecon = r_d312_mred.c12_cvecon
         LET arr_cuentas[vg_cont3].tipnom = r_d312_mred.c12_tipnom

         SELECT *
            INTO r_sn_pen_alim.*
            FROM sn_pen_alim
            WHERE numemp = r_d312_mred.c12_numemp AND
                  cvecon = r_d312_mred.c12_cvecon

         IF STATUS = NOTFOUND THEN
            LET nErrores = nErrores + 1
         ELSE
            LET cCveBene = r_sn_pen_alim.pro_id USING '&&&&&&&&'
--          LET cCveBene = '0', cCveBene[1,6]

            SELECT c18_nombre
               INTO cNomBene
               FROM bd1_sitg:m118_bene
               WHERE c18_cvepro = cCveBene AND
                     c18_tiprov = '4'

            LET arr_penali[vg_cont3].nomesp = cNomBene
            LET arr_cuentas[vg_cont3].cue_id = r_sn_pen_alim.cue_id
            LET arr_cuentas[vg_cont3].cvepro = cCveBene
         END IF

         LET nImporteTot = nImporteTot + r_d312_mred.d12_impor

         SELECT cxp
            INTO arr_cuentas[vg_cont3].cxp
            FROM sn_terceros
            WHERE proceso = cTipoProc AND
                  pernom  = arr_cuentas[vg_cont3].pernom AND
                  nomcal  = arr_cuentas[vg_cont3].nomcal AND
                  numemp  = arr_penali[vg_cont3].numemp  AND
                  cvecon  = arr_cuentas[vg_cont3].cvecon

         IF STATUS != NOTFOUND THEN
            LET vg_cont2 = vg_cont2 + 1
         END IF

         IF vg_cont3 = 200 THEN
            EXIT FOREACH
         END IF

      END IF

   END FOREACH

   DISPLAY vg_cont2, vg_cont3

   CALL SET_COUNT(vg_cont3)

   DISPLAY arr_penali[8].* TO scr_penali[8].* --ATTRIBUTE(REVERSE)
   DISPLAY arr_penali[7].* TO scr_penali[7].* --ATTRIBUTE(REVERSE)
   DISPLAY arr_penali[6].* TO scr_penali[6].* --ATTRIBUTE(REVERSE)
   DISPLAY arr_penali[5].* TO scr_penali[5].* --ATTRIBUTE(REVERSE)
   DISPLAY arr_penali[4].* TO scr_penali[4].* --ATTRIBUTE(REVERSE)
   DISPLAY arr_penali[3].* TO scr_penali[3].* --ATTRIBUTE(REVERSE)
   DISPLAY arr_penali[2].* TO scr_penali[2].* --ATTRIBUTE(REVERSE)
   DISPLAY arr_penali[1].* TO scr_penali[1].* --ATTRIBUTE(REVERSE)

   DISPLAY vg_cont3    TO totemp
   DISPLAY nImporteTot TO totimp


   LET _Usuario = FGL_GETENV("LOGNAME")
   LET _Fecha   = TODAY
   LET _Hora    = CURRENT HOUR TO SECOND

   LET _Referencia = 'PERIODO: ',r_d312_mred.s12_pernom USING '&&&&',', CALCULO:
',r_d312_mred.s12_nomcal
                     USING '&&&',', TIPO: ',cTipoNom,', EMPLEADOS: ',vg_cont3,', IMPORTE: ',nImporteTot

   INSERT INTO db_log:bitacora
      VALUES (0, cSistema, _Usuario, _Fecha, _Hora, 'PPAGPENALI', 'C', _Referencia)

   RETURN vg_cont3

END FUNCTION
```

```
##############################################################################
FUNCTION detalle_penali()
   DEFINE
      nKey,
      nScrRow,
      nRegTot,
      nRegRow           SMALLINT


   INPUT ARRAY arr_penali
      WITHOUT DEFAULTS
      FROM scr_penali.*

      BEFORE ROW
         LET nRegRow = ARR_CURR()
         LET nScrRow = SCR_LINE()
         LET arr_penali[nRegRow].x = ">"
         DISPLAY arr_penali[nRegRow].x TO scr_penali[nScrRow].x

      BEFORE FIELD x
         DISPLAY arr_cuentas[nRegRow].cxp TO FORMONLY.cxp

      AFTER FIELD x
         LET nRegRow = ARR_CURR()
         LET nScrRow = SCR_LINE()
         LET arr_penali[nRegRow].x = " "
         DISPLAY arr_penali[nRegRow].x TO scr_penali[nScrRow].x

         LET nKey = FGL_LASTKEY()
         IF nKey = FGL_KEYVAL("DOWN") OR nKey = ENTER OR nKey = TAB OR nKey = RIGHT THEN
            IF ARR_CURR() >= vg_cont3 THEN
               LET arr_penali[nRegRow].x = ">"
               DISPLAY arr_penali[nRegRow].x TO scr_penali[nScrRow].x
               ERROR " There are no more rows in the direction you are going "
               NEXT FIELD x
            END IF
         END IF

      ON KEY ("CONTROL-E")
         CALL calendar()

   END INPUT

   IF INT_FLAG THEN
      LET INT_FLAG = FALSE
   END IF

   DISPLAY ' ' TO FORMONLY.cxp

END FUNCTION
##############################################################################
FUNCTION cxp_pen_alim()
   DEFINE
      vban              CHAR(1),
      cRefere           CHAR(45),
      cPeriodo          CHAR(4),
      cAnio             CHAR(4),
      cMes              CHAR(2),
      cCXP              CHAR(8),
      nHay,
      i,j,k             SMALLINT,
      r_m109_vums       RECORD LIKE bd1_sitg:m109_vums.*,
      cSubCta           CHAR(4),
      cDescripcion      CHAR(20),
      cMensaje          CHAR(120)


   SELECT f06_fecfin
      INTO dFecha
      FROM c306_prno
      WHERE c06_tipnom = '11' AND
            s06_nomcal = 1    AND
            s06_pernom = nPeriodo


   -- GENERAR LAS CUENTAS POR PAGAR

   LET cPeriodo = nPeriodo USING '&&&&'
   LET i        = cPeriodo[3,4]
   LET cMes     = MONTH(dFecha) USING '&&'
   LET cCXP     = 'T', cPeriodo[1,2], cMes, '*'
   LET cAnio    = '20', cPeriodo[1,2]


   -- ASIGNA LOS VALORES FIJOS DE LA CXP.

   LET r_m109_vums.f09_fecreg = dFecha
   LET r_m109_vums.c09_proyec = '02'
   LET r_m109_vums.c09_anuafe = cAnio
   LET r_m109_vums.c09_mesafe = cMes
   LET r_m109_vums.f09_fecpag = ''
   LET r_m109_vums.c09_tipope = '1'
   LET r_m109_vums.c09_edovum = '02'
   LET r_m109_vums.c09_tiprov = '4'
   LET r_m109_vums.d09_sdovum = 0
   LET r_m109_vums.c09_firmae = '36'      -- director de control de pagos
   LET r_m109_vums.c09_firmar = '30'      -- director general de pagos
```

```
      LET r_m109_vums.c09_firmaa = '29'      -- tesorero
      LET r_m109_vums.c09_tidesc = '06'
      LET r_m109_vums.c09_despol = 'OTROS.'
      LET r_m109_vums.c09_edoafe = 'A'
      LET r_m109_vums.c09_origen = '4'
      LET r_m109_vums.c09_numche = ''


      SELECT COUNT(*)
         INTO nHay
         FROM bd1_sitg:m109_vums
         WHERE c09_numvum MATCHES cCXP

      IF nHay IS NULL THEN
         LET nHay = 0
      END IF

      LET j = nHay

      LET _Usuario = FGL_GETENV("LOGNAME")
      LET _Fecha   = TODAY
      LET _Hora    = CURRENT HOUR TO SECOND

      LET k = MONTH(dFecha)

      BEGIN WORK

      FOR i = 1 TO vg_cont3

         IF arr_penali[i].importe > 0 AND arr_cuentas[i].cvepro IS NOT NULL THEN

            SELECT cxp
               INTO arr_cuentas[i].cxp
               FROM sn_terceros
               WHERE proceso = cTipoProc AND
                     pernom  = arr_cuentas[i].pernom AND
                     nomcal  = arr_cuentas[i].nomcal AND
                     numemp  = arr_penali[i].numemp  AND
                     cvecon  = arr_cuentas[i].cvecon

            IF STATUS = NOTFOUND THEN

               LET j = j + 1

               LET r_m109_vums.c09_numvum = cCXP[1,5], j USING '&&&'
               LET r_m109_vums.c09_cvepro = arr_cuentas[i].cvepro
               LET r_m109_vums.d09_impvum = arr_penali[i].importe

               IF j = (nHay + 1) THEN
                  LET cMsgPolizas = ' Se generaron las cuentas por pagar de la ',r_m109_vums.c09_numvum
               END IF
{
               IF arr_cuentas[i].tipnom = '01' THEN
                  IF (arr_cuentas[i].nomcal > 20 AND arr_cuentas[i].nomcal < 30) OR
                     (arr_cuentas[i].nomcal > 120 AND arr_cuentas[i].nomcal < 130) THEN
                     LET cMensaje = 'PENSION ALIMENTICIA DE LA DIETA DE AGUINALDO  DE ',cAnio,
                                    ' DEL DIP ',arr_penali[i].nomemp
                  ELSE
                     LET cMensaje = 'PENSION ALIMENTICIA DE LA DIETA DEL MES DE ',arr_meses[cMes] CLIPPED,
                                    ' DE ',cAnio,' DEL DIP ',arr_penali[i].nomemp
                  END IF
               ELSE
                  IF DAY(dFecha) = 15 THEN
                     CASE
                        WHEN (arr_cuentas[i].nomcal > 20 AND arr_cuentas[i].nomcal < 30) OR
                             (arr_cuentas[i].nomcal > 120 AND arr_cuentas[i].nomcal < 130)
                           LET cMensaje = 'PENSION ALIMENTICIA DE LA PRIMERA PARTE DE AGUINALDO DE ',
                                          YEAR(dFecha) USING '&&&&',' DE ',arr_penali[i].nomemp
                        WHEN (arr_cuentas[i].nomcal > 60 AND arr_cuentas[i].nomcal < 70) OR
                             (arr_cuentas[i].nomcal > 160 AND arr_cuentas[i].nomcal < 170)
                           LET cMensaje = 'PENSION ALIMENTICIA CORRESPONDIENTE A LA PRODUCTIVIDAD DE ',
                                          arr_meses[k] CLIPPED,' DE ',YEAR(dFecha) USING '&&&&',' DE ',
                                          arr_penali[i].nomemp
                        WHEN (arr_cuentas[i].nomcal > 80 AND arr_cuentas[i].nomcal < 90) OR
                             (arr_cuentas[i].nomcal > 180 AND arr_cuentas[i].nomcal < 190)
                           LET cMensaje = 'PENSION ALIMENTICIA CORRESPONDIENTE AL BONO DE FIN DE A#O DE ',
                                          YEAR(dFecha) USING '&&&&',' DE ',arr_penali[i].nomemp
                        OTHERWISE
                           LET cMensaje = 'PENSION ALIMENTICIA DE LA PRIMERA QUINCENA DEL MES DE ',
                                          arr_meses[k] CLIPPED,' DE ',YEAR(dFecha) USING '&&&&',' DE ',
                                          arr_penali[i].nomemp
                     END CASE
                  ELSE
                     CASE
                        WHEN (arr_cuentas[i].nomcal > 20 AND arr_cuentas[i].nomcal < 30) OR
                             (arr_cuentas[i].nomcal > 120 AND arr_cuentas[i].nomcal < 130)
                           LET cMensaje = 'PENSION ALIMENTICIA DE LA SEGUNDA PARTE DE AGUINALDO DE ',
                                          YEAR(dFecha) USING '&&&&',' DE ',arr_penali[i].nomemp
                        WHEN (arr_cuentas[i].nomcal > 60 AND arr_cuentas[i].nomcal < 70) OR
                             (arr_cuentas[i].nomcal > 160 AND arr_cuentas[i].nomcal < 170)
                           LET cMensaje = 'PENSION ALIMENTICIA CORRESPONDIENTE A LA PRODUCTIVIDAD DE ',
                                           arr_meses[k] CLIPPED,' DE ',YEAR(dFecha) USING '&&&&',' DE ',
                                           arr_penali[i].nomemp
                        WHEN (arr_cuentas[i].nomcal > 80 AND arr_cuentas[i].nomcal < 90) OR
                             (arr_cuentas[i].nomcal > 180 AND arr_cuentas[i].nomcal < 190)
                           LET cMensaje = 'PENSION ALIMENTICIA CORRESPONDIENTE AL BONO DE FIN DE A#O DE ',
                                          YEAR(dFecha) USING '&&&&',' DE ',arr_penali[i].nomemp
```

```
                        OTHERWISE
                           LET cMensaje = 'PENSION ALIMENTICIA DE LA SEGUNDA QUINCENA DEL MES DE ',
                                          arr_meses[k] CLIPPED,' DE ',YEAR(dFecha) USING '&&&&',' DE ',
                                          arr_penali[i].nomemp
                     END CASE
                  END IF
               END IF
}
               SELECT c06_mens
                  INTO cMensaje
                  FROM c306_prno
                  WHERE c06_tipnom = arr_cuentas[i].tipnom AND
                        s06_nomcal = arr_cuentas[i].nomcal AND
                        s06_pernom = arr_cuentas[i].pernom

               LET cMensaje = 'PENSION ALIMENTICIA ',cMensaje CLIPPED, ' DE ',arr_penali[i].nomemp


               SELECT c03_deslar
                  INTO cDescripcion
                  FROM d303_mpyd
                  WHERE c03_tipnom = arr_cuentas[i].tipnom AND
                        c03_cvecon = arr_cuentas[i].cvecon

               IF cDescripcion[1,3] = 'RET' THEN
                  LET cMensaje = 'RETROACTIVO DE ',cMensaje
               END IF

               LET r_m109_vums.c09_conce1 = cMensaje[  1, 30]
               LET r_m109_vums.c09_conce2 = cMensaje[ 31, 60]
               LET r_m109_vums.c09_conce3 = cMensaje[ 61, 90]
               LET r_m109_vums.c09_conce4 = cMensaje[ 91,120]

               INSERT INTO bd1_sitg:m109_vums
                  VALUES (r_m109_vums.*)

               LET cRefere = '000',r_m109_vums.c09_cvepro
               LET cSubCta = arr_cuentas[vg_cont3].cue_id[9,12]

               INSERT INTO bd1_sitg:d110_parv09
                  VALUES (r_m109_vums.c09_anuafe, r_m109_vums.c09_mesafe, r_m109_vums.c09_numvum, 1, 'C',
                          '0000', '00402', cSubCta, '0000', '0000', 'D', cRefere, r_m109_vums.d09_impvum,
                          '02', 'C')

               INSERT INTO bd1_sitg:m150_emis
                  VALUES ('00000000', r_m109_vums.c09_cvepro, '4', '00000000', arr_penali[i].nomesp,
                          r_m109_vums.f09_fecreg, TODAY, '00', r_m109_vums.d09_impvum,
                          r_m109_vums.c09_conce1, r_m109_vums.c09_conce2, r_m109_vums.c09_conce3,
                          r_m109_vums.c09_conce4, r_m109_vums.c09_proyec, r_m109_vums.c09_numvum,
                         'G', '1', '4',
                         '0000000000', '00000000')

               UPDATE d312_mred
                  SET c12_numfol = r_m109_vums.c09_numvum
                  WHERE c12_tipnom = arr_cuentas[i].tipnom AND
                        s12_nomcal = arr_cuentas[i].nomcal AND
                        s12_pernom = arr_cuentas[i].pernom AND
                        c12_numemp = arr_penali[i].numemp  AND
                        c12_cvecon = arr_cuentas[i].cvecon

               INSERT INTO sn_terceros
                  VALUES (cTipoProc, arr_cuentas[i].pernom, arr_cuentas[i].nomcal, arr_penali[i].numemp,
                          arr_cuentas[i].cvecon, r_m109_vums.c09_numvum)


               LET _Referencia = 'NOMINA: ',arr_cuentas[i].tipnom,'-',arr_cuentas[i].nomcal USING '&&&',
                                 '-',arr_cuentas[i].pernom USING '&&&&', ', EMPLEADO: ',
                                arr_penali[i].numemp,', BENEFI: ',arr_cuentas[i].cvepro,', IMPORTE: ',
                                arr_penali[i].importe,', CXP: ',r_m109_vums.c09_numvum

               INSERT INTO db_log:bitacora
                  VALUES (0, cSistema, _Usuario, _Fecha, _Hora, 'PPAGPENALI', 'P', _Referencia)


            END IF

         END IF

      END FOR

      LET cMsgPolizas = cMsgPolizas CLIPPED, ' a la ',r_m109_vums.c09_numvum

      COMMIT WORK
   -- ROLLBACK WORK

   END FUNCTION
   ###############################################################################
```

CONCLUSIONES

En la presente Tesis se expuso el trabajo desarrollado en la Unidad de Sistemas Financieros de la Tesorería General de la Asamblea Legislativa del Distrito Federal.

La importancia de esta área se debe a que tiene a su cargo el buen funcionamiento de los sistemas de cómputo, bases de datos y el resguardo de la información que se genera en las Direcciones Generales de Pagos y Presupuesto de la Tesorería General, así como coadyuvar a los usuarios a mejorar y optimizar sus labores dentro de la organización apoyados en la tecnología.

Debido a lo anterior, me significó una excelente oportunidad para aplicar los conocimientos que adquirí en la Facultad de Ingeniería de la UNAM, en el Desarrollo de Sistemas y las Bases de Datos, demostrando mi capacidad para negociar, analizar, evaluar, diseñar, desarrollar, operar y dar soporte técnico a los módulos del sistema para el procesamiento de datos en las diversas áreas de la Tesorería General.

Durante el desarrollo de esta Reingeniería de Sistemas, me di cuenta de la importancia que tiene para la Institución contar con un sistema confiable y expedito que le permita mantener los datos bien organizados y a buen resguardo, pudiendo efectuar un mejor análisis de la información que de ellos emana y proporcionar en tiempo y forma los informes que sean requeridos por las autoridades superiores.

La presente muestra la factibilidad de llevar la Reingeniería de Sistemas a un sistema administrativo que había llegado a ser solo un repositorio de datos (en el módulo de nómina) y que eran cargados desde una hoja de cálculo en la que se efectuaban los cálculos correspondientes y explotados posteriormente únicamente con algunos reportes existentes.

El haber hecho la Reingeniería de Sistemas al SITG, redujo radicalmente el tiempo para procesar las nóminas, los pagos en general y la contabilidad, permitiendo que los usuarios retomaran su horario laboral normal y fuesen más productivos.

Encontré que para que un proyecto llegue a buen fin en un corto tiempo es muy importante que exista un compromiso real entre las partes involucradas, tanto del lado del usuario como del lado del equipo de Ingenieros de Sistemas.

Desde el momento en el que se liberaron los procesos de la prioridad 1, los usuarios finales constataron que su trabajo se redujo notablemente y su productividad aumentó, evitando los errores y el tiempo en la captura de datos que se tenían al cargarlos en un módulo y que eran generados en otro.

Lo presentado en esta Tesis es una pequeña muestra de todo lo que se desarrolló para el SITG. Se crearon muchas nuevas tablas para el módulo de nómina, se reajustaron todos los reportes burdos para ser impresos en impresoras láser, se desarrollaron programas para respaldar y recuperar las bases de datos, se desarrollaron programas en Delphi para imprimir en formas preimpresas y con formatos específicos y se hicieron consultas no planeadas en MSQUERY.

Demostré que es posible no sólo reajustar los programas que requieren ser modificados y readaptados a los nuevos requerimientos y crear nuevos a partir de prototipos en el mismo ambiente en poco tiempo, sino que incluso sería posible convertirlos a una nueva plataforma con un nuevo enfoque, también en un lapso relativamente corto, aprovechando la Evaluación de procesos para extraer los algoritmos importantes o valiosos que se reutilizarían en la nueva plataforma.

BIBLIOGRAFÍA:

Ingeniería de Software – Un enfoque práctico– Quinta edición – Roger S. Pressman
ISBN 84-481-3214-9 McGraw-Hill
www.mcgraw-hill.es

Ingeniería de Software – Un enfoque práctico– Sexta edición – Roger S. Pressman
ISBN 970-10-5473-3 McGraw-Hill
www.mcgraw-hill.es

Ingeniería de Software - Richard E. Fairley
ISBN 968-451-854-4 McGRAW-HILL
QA766 F3518 1987 G-134738

Fundamentals of Software Engineering – Carlo Ghezzi/Medí Jazayeri/Dino Mandrioli
ISBN 0-13-820432-2 PRENTICE HALL
QA76.7S8 G44 1991 G.-141284

Software Engineering 6th Edition – Ian Summerville
ISBN 0-201-39815-X Addison-Wesley
QA76.758 S65 2000 G.-170583
www.pearsoneduc.com

Ingeniería de Software Segunda Edición – Ian Summerville
ISBN 968-6135-91-X Addison-Wesley Iberoamericana

Relational Database Design – 1996 Informix Software, Inc.
Part No. 000-4829
Book No. 502-54331-999999-1
Versión 01-96