UNIVERSIDAD NACIONAL AUTONOMA DE MÉXICO
ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES
CAMPUS ARAGÓN

ANÁLISIS, DISEÑO Y DESARROLLO DEL SISTEMA DE RECURSOS HUMANOS PARA LA SECCIÓN AMARILLA

TESIS
QUE PARA OBTENER EL TITULO
INGENIERO EN COMPUTACIÓN
PRESENTA:

EDGAR HERRERA CASAS

ASESOR DE TESIS: ING ERNESTO PEÑALOZA ROMERO

México 2004

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES ARAGÓN

JEFATURA DE INGENIERÍA EN COMPUTACIÓN

OFICIO: ENAR/JACO/0005/04

ASUNTO: Designación de Revisores.

ING. ROBERTO BLANCO BAUTISTA

M. EN C. JESÚS DÍAZ BARRIGA ARCEO

ING. ERNESTO PEÑALOZA ROMERO

ING. IMELDA DE LA LUZ FLORES DÍAZ

ING. RAFAEL CANTO GALLO R canto G.

Informamos a ustedes de la autorización que se le concede al alumno **EDGAR HERRERA CASAS,** para que puedan desarrollar el trabajo de tesis titulado: **"ANÁLISIS, DISEÑO Y DESARROLLO DEL SISTEMA DE RECURSOS HUMANOS PARA LA SECCIÓN AMARILLA"** dirigido por el **Ing. Ernesto Peñaloza Romero,** solicitando a ustedes sean tan amables de revisar el avance del mismo y hacer las observaciones que consideren pertinentes, o en su caso, indicar al alumno si dicha revisión se hará a la conclusión del trabajo de tesis.

Sin otro particular, me es grato enviarles un cordial saludo.

A T E N T A M E N T E
"POR MI RAZA HABLARA EL ESPIRITU"
San Juan de Aragón, Estado de México, 6 de enero del 2004.
EL JEFE DE CARRERA

M. EN C. JESÚS DÍAZ BARRIGA ARCEO

JDA*vjd

AGRADECIMIENTOS

A Dios mi Señor:

Por darme la fortuna de poder conocerlo
y las herramientas espirituales y materiales
necesarias para haber realizado este trabajo,
y también por darme todo.

A mis Padres Salvador y Estelita:

Por ser las personas que me amaron antes
de mi nacimiento, y por sus cuidados conmigo
desde pequeño, sin ellos esto no hubiera sido
posible este trabajo, y por todo ese Amor
que he sentido siempre ...

A mis Hermanos Marisol, Lucero y Salvador:

Por estar siempre conmigo apoyándome en
mis decisiones con todo su amor.

A mis Amigos Francisco, Isabel, Gloria, Adriana, Susana, Elizabeth:

Que tan especialmente me han brindado su amistad
sin pedir algo a cambio.

A mis Maestros de toda la carrera:

Que me han guiado en el sendero del conocimiento,
Y me han enseñado que camino es aquel que yo
mismo hago..

RESUMEN DEL INDICE

INDICE

Introducción

El área más importante de una empresa es Recursos Humanos, podríamos hablar mucho de otras como, producción, ventas, de aquellas de las cuales depende la empresa para tener un ingreso y cumplir su objetivo, sin embargo, para que cada una de estas áreas cumplan su objetivo, se necesita del trabajo de la gente, y aquí es donde llegamos al punto, a la columna vertebral de cualquier empresa, la gente, por mas robots y automatización que existan, siempre se necesitará de la gente, y aquí es dónde entra Recursos Humanos, esta área es la que se encarga de ver por todas las personas pertenecientes a la empresa, desde la gente de intendencia hasta el director general. El buen funcionamiento de una empresa comienza en su gente, por lo tanto el tiempo que se les dedique es vital, entonces toda su administración su bienestar significa el bienestar, crecimiento, y desarrollo de la empresa, empezamos a entender la importancia que tiene el departamento de recursos humanos. Existen muchas etapas[1] en este departamento, una de ellas es el proceso de ingreso a la empresa como empleado, esto implica muchas cosas, comienza por la etapa de reclutamiento, aquí es dónde podemos encontrar a los posible candidatos a las plazas libres, para tener a estos se recurre a diferentes planes de promoción de la plaza, como la inscripción a la empresa en algún portal de bolsa de trabajo de Internet, los acuerdos con otras empresas de intercambio de candidatos, publicaciones en diferentes diarios, etc. Una vez que se tienen a los candidatos el primer paso es llenar la solicitud estándar o propia de la empresa, en dónde el candidato escribirá todos sus datos personales, las claves de sus documentos oficiales, las instituciones a las que pertenece, datos familiares, nivel profesional, habilidades personales y trabajos anteriores, esta es la primer información que obtiene la empresa acerca del candidato, esta información es analizada por el auxiliar de reclutamiento, que en primera instancia decidirá en base a la información obtenida, si se continúa el proceso o se mantiene en cartera al candidato, o bien queda descartado desde la primer etapa y talvez no sea apto ni para intercambiar la información de este candidato con otras empresas. De lo contrario continúa el proceso, ahora cada empresa tiene una serie de exámenes los cuales nos proporcionarán información de índole psicológica, de habilidad mental, de comportamiento, y de creencias entre otras, toda esta información es analizada, calificada y comparada con resultados preestablecidos, en base a esto se vuelve a tomar la decisión, si se continúa el proceso o no.

Si es que se continúa el proceso, el candidato pasa a la siguiente etapa que se trata de los exámenes de conocimientos, cada empresa puede tener sus exámenes ya establecidos para cada área, estos son evaluados por el experto en el tema, puede ser el gerente el director o jefe de área el cuál regresa un informe a Recursos Humanos dónde se certifica si el candidato es apto para el puesto solicitado de lo contrario se guardaría en cartera. El siguiente paso son los exámenes médicos en las instancias médicas que la empresa señale o las que el candidato elija, los resultados de estos últimos también influenciará en la decisión de la continuación del proceso. Por último se realiza un examen socioeconómico al candidato, de este examen por lo regular se encarga una empresa externa contratada, pero es probable que lo realice la misma empresa. Concluidos los exámenes satisfactoriamente se procede al ingreso del nuevo empleado, pidiéndole todos los documentos necesarios como acta de nacimiento, RFC , CURP cartas de recomendación, fotografías, comprobante de domicilio, los cuales son necesario capturar de manera legítima es decir su almacenamiento como imagen, se le asigna una clave (no en todas las empresas) y con todos estos documentos y los que genere la propia empresa como contrato, alta del IMSS, AFORE, Seguro de Vida, dónde también se guardará permisos y observaciones, se integrará el expediente del empleado, dónde en cualquier momento y por las personas autorizadas podrá ser consultado. Así como también se le asigna un salario diario y en base a éste y todas las prestaciones que ofrezca la Empresa se calcula su salario integrado que es el que se toma en cuenta para la presentación de todas las declaraciones requeridas por el IMSS. A partir de aquí el empleado comenzará una historia de movimientos de percepciones, deducciones, salarios normales, salarios integrados, movimientos del IMSS, cambios de puesto y acumulados de las prestaciones correspondientes, por lo tanto la información del empleado comienza a fluir por todo el departamento, ya que entrará en un organigrama de la empresa en el nivel que le

[1] Se refiere a un cierto nivel de avance en un proyecto con una organización establecida.

corresponde así como en la nómina, prestaciones que le brinda la empresa, como también obligaciones que se le exige.

Ahora mencionamos una parte fundamental del área de Recursos Humanos, La Nómina, implica muchas cosas, no sólo el cálculo de las percepciones, deducciones, liquidaciones, aguinaldos, fondo de ahorro, aumento de salarios, reparto de utilidades de cada empleado, si no también se realizan declaraciones de impuestos y generación de nuevas prestaciones y control de faltas, incapacidades, permisos y todas las incidencias[2] existentes en la empresa, así como una consulta detallada y constante de todos los movimientos actuales y pasados, con años de anterioridad, realización de pólizas contables y una gran gama de reportes específicos y variables que se utilizan. Por todo lo anterior se necesita una auténtica administración de históricos y una integridad imperativa para los cálculos que los utilicen. Alguna nóminas utilizan salarios fijos, otras tienen salarios variables, es decir el salario de cada empleado es variable a causa de su actividad, la cual puede ser comisionista, aquí es dónde debemos de tener mucho cuidado por que esto implica una gran variabilidad en las percepciones y deducciones, si es que en estas basamos todos los cálculos, por que una quincena puede salir muy elevada y es cuando puede pagar prestamos o deducciones como INFONAVIT, o puede pasar lo contrario, una quincena es tan reducida que las percepciones resultaron menores que las deducciones, esto indica un resultado negativo, y según las políticas de cada empresa se tratará este resultado. Por eso es necesario tener la facilidad de modificar los resultados finales después de haber pasado por todos los filtros correspondientes como topes y prioridad de descuentos establecidos..

En el departamento de Recursos Humanos es probable que se encuentre Capacitación, la cual se encarga de realizar un estudio detallado del conocimiento por parte de los empleados en el campo que se están desarrollando dentro de la empresa, por lo tanto es importante tener una total administración de los cursos propuestos para cubrir las carencias de conocimientos detectadas en las diferentes áreas, así como también el estudio de nuevas metodologías y tecnologías que van surgiendo.

Por lo tanto en el Departamento de Recursos Humanos encontramos que se necesita una total administración de personal, entonces se tiene que recurrir a una estrategia para la organización de la información. Un análisis Entidad – Relación sería conveniente, de esta forma podemos definir los datos de mayor jerarquía, aislando aquellos que no se ven muy involucrados en el proceso

[2] Permisos, faltas, incapacidades, son conceptos con valores constantes, clave para el cálculo de otros conceptos.

Capítulo I
Marco Teórico

En el Análisis y Diseño de un Sistema de Información se requiere pasar forzosamente por varias etapas, estas en muchas ocasiones son ignoradas, ya que no se cuenta con el conocimiento de ellas o simplemente por que se decidió no pasar por ellas. Sin embargo al ignorar estas etapas se incurre en un grave error, el cual se verá reflejado durante el desarrollo del sistema y la implementación, ni hablar cuando se encuentre en producción, es ahí dónde los problemas graves se presentan a todas luces, no pretendo explicar cada una de estas etapas detalladamente solo mencionaré las mas importantes, y el porqué no se pueden ignorar.

1.1. Análisis, Diseño y Desarrollo de un Sistema de Información

Al pasar del tiempo todas las personas se han dado cuenta de la necesidad de organización, esta palabra involucra muchas cosas, ya que a partir de que las empresas han ido creciendo, su volumen de información ha aumentado, información sobre todas las personas que laboran en la empresa, sobre las materias primas, productos terminados, ventas, declaraciones, etc. La diferencia en llevar esta información en papel a un almacenamiento en una computadora es abismal, ya que en ésta última, podemos acceder a la información mas rápidamente y elegir gestores por módulos, existe software que ya realiza esto, son paquetes comerciales que las compañías de desarrollo lanzan a la venta, sin embargo, estos paquetes están hechos de una manera muy generalizada y básica, es decir, la eficiencia de estos es muy limitada porque no pueden cubrir todos los requerimientos de cada empresa, esto nos lleva a la necesidad de desarrollar un producto ideal para cada empresa, entonces es cuando surge de manera particular el Análisis, Diseño y Desarrollo de un Sistema de Información especializado, o bien un sistema que cumpla con los requerimientos generales, y este se personaliza para cada empresa.

Veamos de una manera general las etapas por las que debe pasar un Sistema de Información (Fig 1.1), las cuales no se llevan a cabo separadas, es decir, se pueden hacer algunas en forma simultánea, así en un cronograma podemos reducir tiempos.

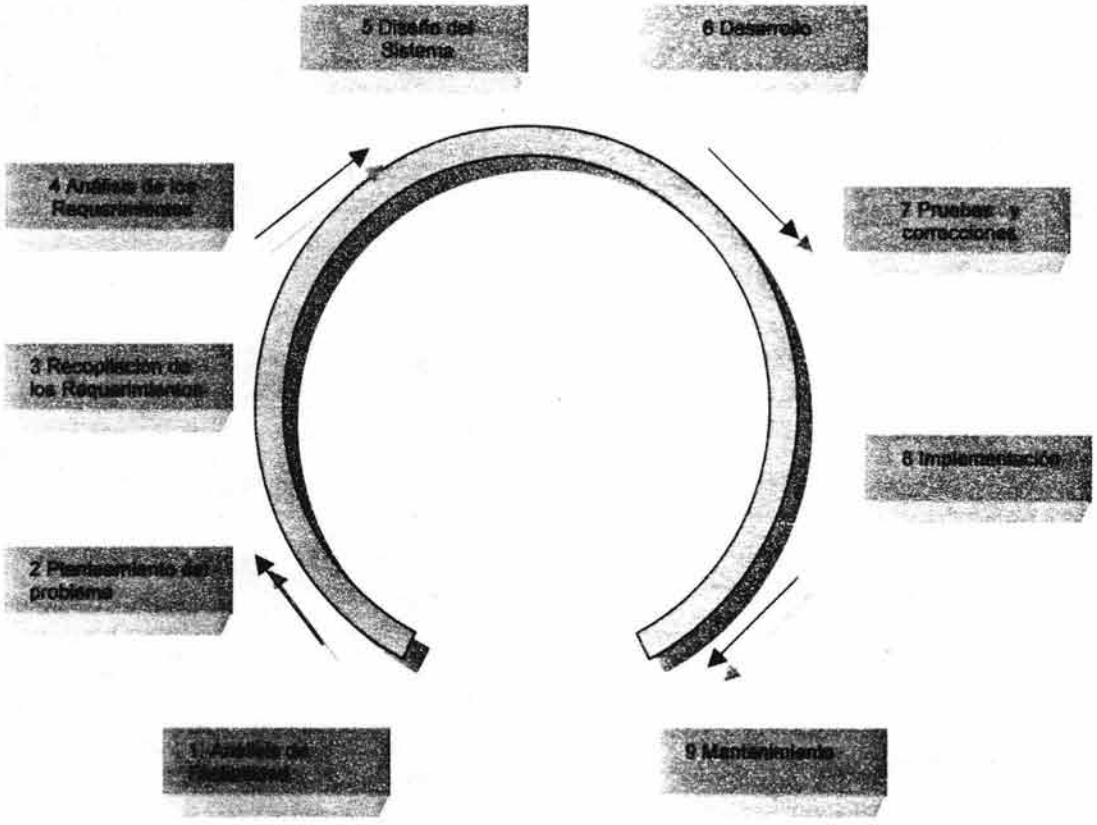


Figura 1.1
Ciclo de vida de un Sistema de Información

1.1.1 Estudio preliminar – Análisis de factibilidad

Esta es la primera etapa en la que se trabaja durante el ciclo de vida de un sistema. El estudio preliminar y análisis de factibilidad son aquellos que describen la situación actual en cuanto a recursos existentes en software y hardware en la empresa Este estudio se basa también en el presupuesto destinado.

Entonces, el estudio preliminar junto con el análisis de factibilidad nos puede dar diferentes respuestas:

- Comprar un producto comercial y no desarrollar un nuevo sistema.- Investigar si algún software existente en el mercado cubre nuestras necesidades, así nos ahorraremos el tiempo de desarrollo, y la empresa invertirá en el costo del producto, y la capacitación correspondiente. Cuando se tiene esta respuesta es por lo general que el equipo de sistemas es de la propia empresa, y trabaja al 100% por los intereses de la empresa. Comprar software comercial puede ser menos costoso que desarrollarlo, ya que este estudio no lo realiza un consultor, el solo presentará su propuesta.
- Actualizar el sistema actual agregando módulos o utilerías y no desarrollar un nuevo sistema. - Esta solución la obtenemos a partir de un estudio al sistema actual, tomando en cuenta la tecnología utilizada por el mismo, si es que tenemos total acceso a él, y si es flexible para desarrollar mas módulos, ampliando por así decirlo la vida del sistema actual. Esta decisión pasa por lo general en los sistemas pensados para su escalabilidad, es decir un sistema puede caminar junto con la empresa durante un tiempo considerable. En este caso talvez así sea más rentable, sin embargo hay que tener mucho cuidado cuando se toman estas decisiones, ya que si nos equivocamos, podemos caer en el hecho de no hacer el sistema, puede salir mucho más costoso que si lo hubiéramos hecho. Debemos recordar que el análisis de factibilidad lo realiza el analista de sistemas para presentarlo a los jefes administrativos, se presentan los costos y tiempos, y así ellos tomarán una decisión.

- Desarrollar un nuevo sistema.- Esta es la decisión que a nosotros nos interesa, ya que pretendemos desarrollar un sistema completo, y es aquí dónde continúan todas las etapas del ciclo de vida de un sistema.

1.1.2 Planteamiento del problema

En esta etapa se encuentra la piedra angular de lo que será nuestro sistema, la identificación del problema nos viene a trazar un camino, el cual recorreremos durante todo el desarrollo, esta parte es decisiva en el éxito o fracaso del sistema, ya que si no se identifica correctamente el problema, estaremos dando vueltas y vueltas resolviendo un problema que tal vez ni existe, puede oírse muy descabellado, sin embargo a sucedido, y efectivamente ha sido el fracaso del sistema, pérdida de tiempo y dinero, sin mencionar el daño a nivel empresa. En el planteamiento del problema participan desarrolladores, administradores y usuarios del sistema, que al fin y al cabo tienen identificado cuál es el problema, es por eso que se ha requerido el desarrollo del sistema. Sin embargo el equipo de sistemas tiene que conceptual izarlo para saber qué método se utilizará para abordar dicho problema.

Las entrevistas con los usuarios son extremadamente importantes, las cuales se deben de estructurar, de manera que se traten los puntos claves de todos los problemas existentes, es decir, una entrevista se debe de planear; Kendall & Kendall en su libro "Análisis y Diseño de Sistemas", proponen cinco pasos para la planeación de una entrevista, estos se muestran en la Figura 1.2.

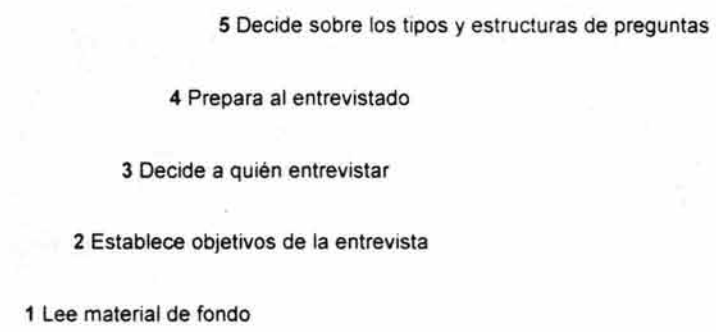


Figura 1.2
Cinco pasos para la planeación
de una entrevista según Kendall & Kendall

Las primeras entrevistas se realizan con el fin de obtener un panorama general de los procedimientos y problemas existentes en la empresa, realizar una lista de ellos y así presentar los objetivos globales.

1.1.3 Recopilación de los Requerimientos

En esta etapa se profundizará en todos los requerimientos, los cuales deben de ser totalmente documentados, en donde se plasmará detalladamente cada procedimiento y regla de negocio existentes, así como todos los conceptos de la propia empresa. Para lograr esto tenemos que identificar a todos los usuarios involucrados en los procesos correspondientes, como también a todas aquellas personas de los departamentos relacionados con dichos procesos, lo anterior se hace por medio de la observación y planteamiento de frases como:

¿A quién observar?
Se deben identificar a todos aquellos usuarios que serán afectados por el sistema.

¿A quién preguntar?
Sucede muchas veces que los analistas se dirigen al jefe de personal, director, etc, es cierto que son necesarias las entrevistas con dichas personas, sin embargo todos los operadores se encuentran totalmente empapados de los procesos y son a ellos a quiénes nos debemos de dirigir, también sucede que dentro de estos usuarios hay gente mas experta en la materia, entonces estos son nuestros usuarios expertos, ya que debemos hacer buena relación con ellos porque son las personas que más enriquecerán a nuestro sistema, así se reduce la posibilidad de no cubrir algún requerimiento.

¿Qué examinar?
Existen muchos reportes, formatos, documentos de salida, algunos son generados y otros son requeridos por las instancias gubernamentales, es posible que estos cambien cada año o cada semestre, es por eso que se debe de hacer un estudios sobre cuáles documentos vamos a tomar en cuenta, y cuáles vamos a ignorar.

¿Qué solicitar?
Aunque el usuario se encuentre totalmente dispuesto a dar la información deseada, hay procesos que talvez no nos diga, esto puede ser porque simplemente se le olvidó en ese momento, o es celoso de su trabajo, el analista aparte de hacer una buena relación con el usuario para que toda la información sea proporcionada en un ambiente de confianza, también debe de preguntar sobre posibles procesos no mencionados de acuerdo a las experiencias del analista en sistemas similares ya desarrollados, sin embargo como cada empresa tiene sus propias reglas de negocio, no es imposible adivinar en su totalidad todos aquellos procesos y reglas que no nos mencione el usuario, por lo tanto es recomendable estipular en un documento lo proporcionado por el usuario, ya que en un momento dado si se detecta que no se cubrió algún requerimiento, se muestra toda la información que proporcionó e usuario.

Para una buena recopilación de datos es necesario utilizar el muestreo, es decir, no es necesario entrevistar a todos los operadores o a todos los ejecutivos, o monitorear todos y cada uno de los elementos existentes en un proceso, esto sería muy costoso, entonces para evitar esto se toman muestras de cada población, así el comportamiento será igual para la producción en serie, sin embargo se debe tener en cuenta la medida total de la producción, es decir, nunca debemos de perder de vista el tamaño de la producción y en sí el tamaño de la empresa.

Tipos de información

Antes que nada en una entrevista se debe preguntar si existen ya manuales de procedimientos, estos nos ayudarán a reducir las entrevistas y documentarnos nosotros mismos, así también lograremos hacer preguntas mas concretas, ya que se invierte mucho tiempo en preguntas abiertas. Identificar los tipos de información que podemos recolectar, nos ayuda a organizarnos, por lo general existen: Manuales de procedimientos, Reportes, Documentos oficiales, Documentos de captura, Memorándums, etc, estos últimos nos

proporcionarán información sobre la actitud de los gerentes y el ambiente que se respira en la empresa como también el uso que se le da a la información y las creencias que existen en la empresa.

La observación

La observación en el desarrollo de un sistema es extremadamente importante, de ahí pueden surgir muchas cosas, se debe de observar el comportamiento del tomas de decisiones, estando en los niveles mas altos de la empresa, podemos enteramos cómo es que se manipula la información, incluso obtener algunos requerimientos que ni ellos mismos saben que los necesitan, lo mismo sucede con los operadores. Por ejemplo, un operador es entrevistado, responde todas las preguntas que se le hacen, sin embargo al estar hablando como no tiene mucho tiempo, también está realizando su trabajo en pequeñas pausas, el analista observando ese trabajo se da cuenta que esta haciendo todo un reporte a mano, el cuál no le fue mencionado, el analista hace una pequeña nota para preguntar por ese reporte mas adelante, para no interrumpir al operador.

Es importante que las entrevistas sean breves, ya que los ejecutivos y operadores se encuentran haciendo su trabajo y él atendernos es un tiempo extra de inversión para ellos, y si la entrevista se toma un porco larga, el usuario comenzará a desesperarse por su atraso en el trabajo, y la próxima vez que queramos entrevistarlo no será confortable para él, y es muy probable que no nos proporcione toda la información que necesitamos.

Prototipo

El desarrollo de un prototipo no ayuda a obtener los requerimientos más específicos, este nos ayudará a mejorar el desarrollo del sistema y como consecuencia los resultados. El usuario al ver un prototipo de lo que va a ser el sistema nos puede proporcionar más requerimientos incluso aclarar algunos que tal vez no fueron comprendidos en un 100%. El usuario también nos puede ayudar en el diseño de la interfaz, ya que es la persona que operará el sistema nos informa de qué manera se le facilitará más el manejo del mismo.

Es recomendable hacer un prototipo en los casos de que se tenga la idea de desarrollar un sistema totalmente nuevo, no se tiene que hacer el prototipo de todo lo que será el sistema, ya que de algunos módulos no será necesario. Un prototipo es funcional aunque no tendrá todas las funciones de lo que será el sistema, tampoco contará con todas las características, si no que tendrá aquellas más representativas.

El prototipo se realizará siempre y cuando se cuente con el presupuesto y el tiempo, de lo contrario puede ser contraproducente y robamos tiempo de desarrollo para el sistema.

1.1.4 Análisis y Diseño de un Sistema de Información

Comenzamos a analizar la información proporcionada por el usuario, la cual es necesaria conceptualizarla y exponerla de una manera gráfica, así tendremos una visión más cristalina de los procesos. Esto lo haremos mediante Diagramas de flujos de datos, dónde podemos plasmar absolutamente todos los procesos y reglas de negocio por módulos, también nos ayudará a la interpretación técnica sin que profundicemos en ella, ya que lo importante ahora es tener bien entendido el flujo de datos sin preocuparnos por el diseño técnico todavía. El diagrama de Flujo de Datos será presentado a los usuarios, con el fin de que hagan comentarios sobre los mismos, y así los analistas tendremos la certeza de que se han conceptualizado bien los procesos, o bien se realicen cambios según las demandas de los usuarios, este es el momento ideal para hacerlos, antes de interpretarlos en la computadora. Para que suceda esto correctamente, el analista debe de expresarse con toda claridad, es decir, el usuario debe de entender en su totalidad la conceptualización que tienen los analistas del sistema y así evaluarlos, es decir; "Estamos todos de acuerdo".

1.1.4.1 Diagrama de Flujo de Datos

Como ya mencioné es una manera más transparente de ver el flujo de información, existen convenciones para los diagramas de flujos de datos, muchos autores manejan varios símbolos, sin embargo no recomiendo utilizar mas que los cuatro símbolos básicos de un diagrama de flujo, que son; Entidad, Flujo de datos, Proceso y Almacenamiento de datos, es tos los podemos ver en la Figura 1.3.

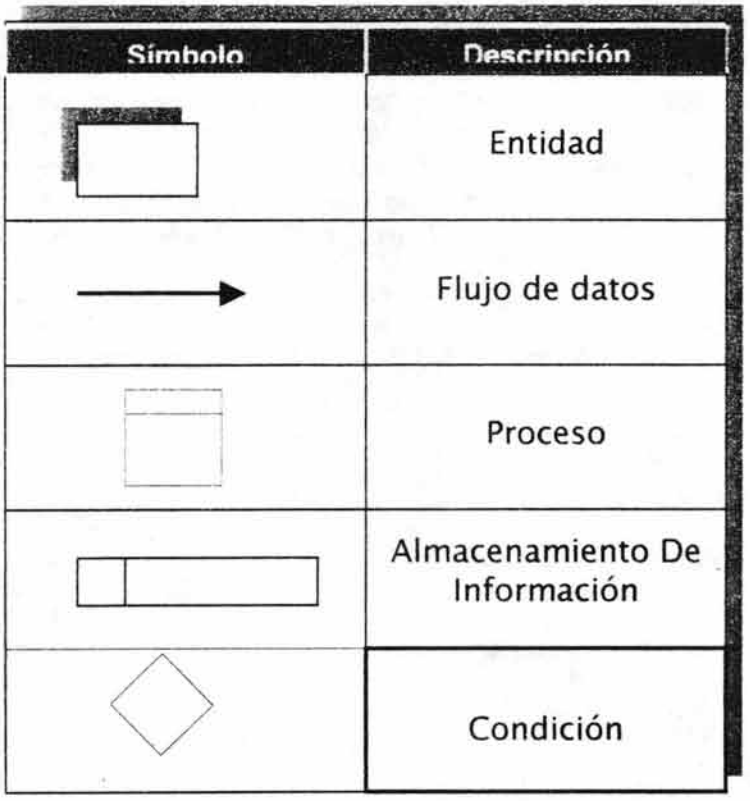


Figura 1.3
Símbolos de un Diagrama de Flujo

De los símbolos de un diagrama de flujo de datos, la Entidad puede ser un elemento externo, una persona, una máquina, un departamento que envía información o bien la recibe, es etiquetada por un nombre. La Flecha indica el flujo de información, hacia a dónde va la información y de dónde viene esta. Proceso indica que es un proceso interno, una caja negra donde existe una serie de operaciones y dónde se transforma la información. Almacenamiento de información indica simplemente un contenedor de información, no especificaremos que tipo de contenedor ya que puede ser un disco duro, una unidad de cinta, etc.

Lo primero que hay que hacer para desarrollar un Diagrama de Flujo de Datos es identificar todos los componentes como son; las entidades, los procesos y los almacenamientos de información. Sin embargo es necesario hacer lo que llamarán Kendall & Kendall, un diagrama de contexto, dónde identificaremos a los módulos globales, las entradas y salidas básicas del sistema, y así obtendremos un panorama general del sistema, aquí planteamos a todo el sistema en un solo proceso, Figura 1.4.

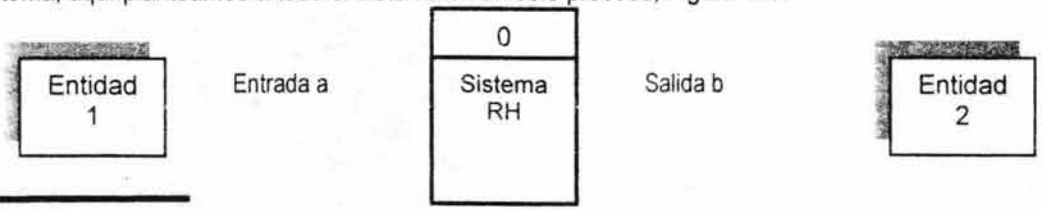


Figura 1.4
Diagrama de contexto

Ahora veremos un ejemplo de cómo desglosaremos el diagrama anterior en un diagrama de flujo de datos completo, es decir, estamos presentando a todo lo que será el sistema en un solo proceso, especificando una única entrada y una única salida, pues bien, abramos esta caja negra para ver lo que tiene por dentro, puede contener n procesos, n condiciones, n almacenamientos de información y n entidades, ya que estamos retomando la metodología de lo general a lo específico, pienso que es la más recomendada ya que teniendo un panorama general del universo que vamos a manipular, es más fácil que identifiquemos poco a poco las entidades y procesos específicos por módulos, y esto lo hacemos cada vez que profundizamos más y más, hasta llegar a los elementos básicos de nuestro sistema. Ver Figura 1.5. En este diagrama podemos observar que existe una entidad, y es la selección de los aspirantes a entrar a cierta empresa por parte de los auxiliares de personal, posteriormente tenemos un proceso dónde se incluyen todos los exámenes que realizará los aspirantes, desde exámenes psicológicos, psicométricos hasta técnicos y estudios socioeconómicos, después se procede a almacenar los resultados de los exámenes, esto se encuentra en el almacenamiento "D1", en el proceso 1, se califican todos esos exámenes y entra a una condición, si el aspirante es aceptado de acuerdo al perfil de puesto y a los resultados de los exámenes, entonces entra al proceso 2 que es el proceso de ingreso a la empresa, que incluye todos los trámites, por otro lado si no pasa los exámenes se almacena en una cartera de aspirantes.

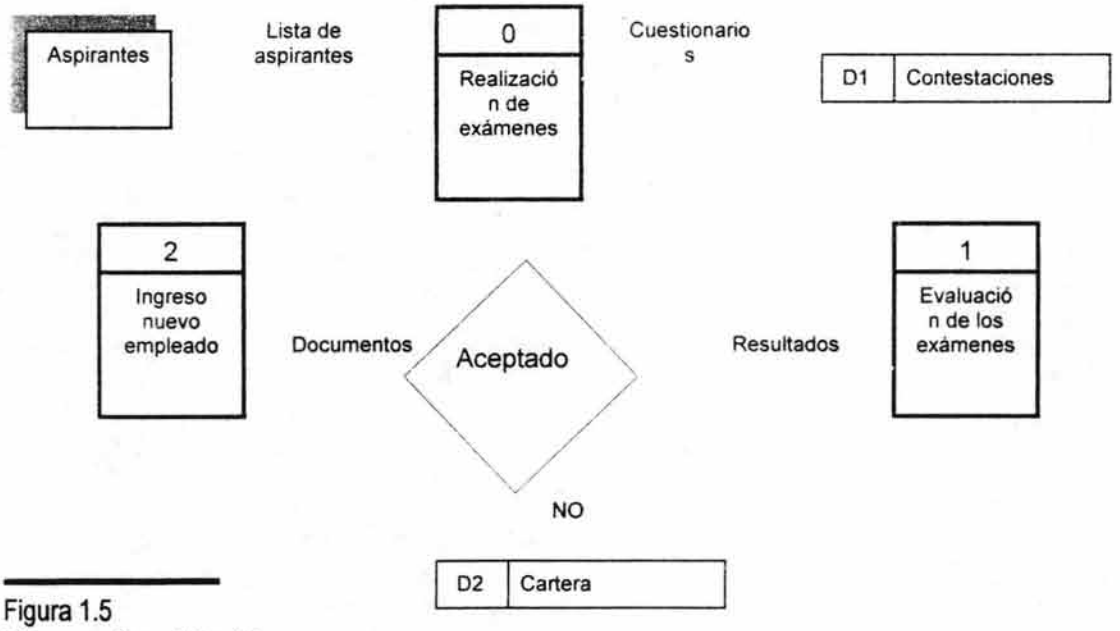


Figura 1.5
Diagrama Completo del Sistema RH

Vamos a profundizar en nuestro sistema, detallándolo más, explotando al proceso 2 del diagrama principal, en un diagrama hijo (Ver figura 1.6) , en el cual vamos a numerar a sus procesos como 2.1, 2.2, etc., ya que el diagrama principal es el proceso 0, recordémoslo en el diagrama de contexto.

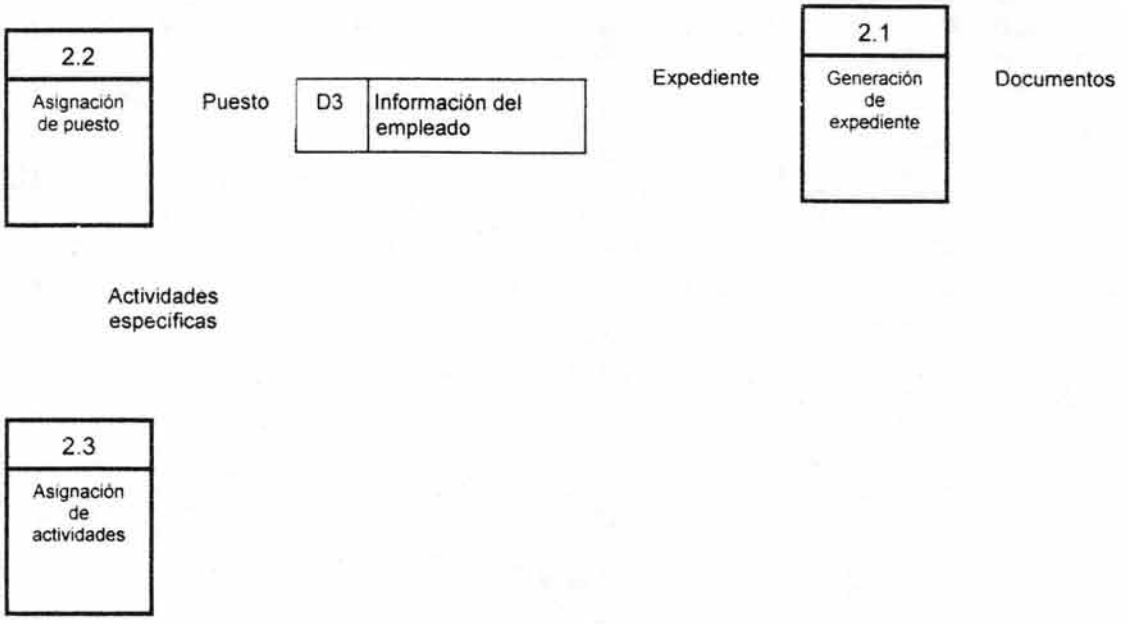


Figura 1.6
Detalle del proceso 2

1.1.4.2 Diccionario de datos

El diccionario de datos no es mas que referencias a los mismos datos, estos son llamados meta datos, y nos ayuda a los analistas a entender mejor los conceptos que se estarán manejando en el sistema, podemos partir de los diagramas de flujo de datos, también nos ayudará en gran medida en el diseño de nuestra base de datos (se verá en el capítulo IV en el diseño de la base de datos). Utilizando el diagrama de flujo de datos de la figura 1.6 realizaremos su estructura de datos en base al flujo de los mismos.

Documento =	Nombre del documento+ Clave del documento+ Fecha del documento+ Tipo de documento
Tipo de documento =	[Oficial | No oficial]
Expediente =	Número de expediente + Fecha de expediente+ Número de empleado+ Nombre de empleado+ Dirección+ Teléfono+ Sexo+ Fecha de nacimiento+ Fecha de ingreso
Nombre de empleado =	Nombre+ Apellido paterno+ Apellido materno
Puesto =	Número del puesto+ Nivel del puesto Sueldo
Actividad =	Número de la Actividad Descripción de la actividad

1.1.4.3 Análisis y Diseño Orientado a Objetos

Anteriormente los problemas se resolvían de una manera estructurada y de una lógica secuencial, ahora es diferente, los problemas planteados se transportan a lo que es llamado "Diseño orientado a objetos", esto se realiza por medio de una abstracción de la realidad, la mejor manera de resolver problemas es transportándolos a una forma diferente de visualización, y a un cambio de ambiente, sin alejarse del problema, esto no es sencillo, sin embargo es muy eficiente, ya que nos brinda muchas ventajas una vez terminado el diseño. Al hablar de un "Diseño orientado a objetos" me estoy refiriendo tanto en aspecto de la información como la manipulación de la misma, es decir, la Base de Datos y la programación.

En los problemas de hoy, nos encontramos con una gran variedad de cambios que se realizan en las empresas, esto es consecuencia del desarrollo incremental de las mismas, esto significa que existen cambios continuos en el software utilizado, por lo tanto no se puede pensar, como antes se hacía, en sistemas duros, en aquellos programados solo para resolver el problema que nos aqueja en ese momento, si no que hoy día con el análisis y diseño orientado a objetos pensamos en una gran flexibilidad de resolución de problemas, es decir, un tanto cuanto predictivos, ciertamente no sabremos qué problemas se nos presentarán en unos años, o incluso en unos meses, sin embargo, con el análisis y diseño O – O [3] nos resulta mas sencillo reutilizar código fuente, esto significa que cuando llegue el momento de realizar una modificación al sistema, esta no tenga que ser tan engorrosa, lo que pasaría en un sistema programado de manera estructurada.

[3] Orientado a Objetos

En la Ingeniería de Software el análisis y diseño O – O vino a cambiar toda una perspectiva de los problemas planteados, como se menciona en el libro Análisis y Diseño de Sistemas de Kendall & Kendall: "La programación O – O toma su concepto de encapsulación de la idea de ingeniería de software de la abstracción de datos, y su concepto de herencia a partir de la idea de base de datos de generalización y especialización.",

1.1.4.3.1 Análisis Orientado a Objetos

Antes se pensaba en la resolución de problemas, atacándolos directamente, es decir, se creaban módulos que se dedicaban a una tarea específica, y los cuales no eran muy dinámicos, hoy en día con el modelado orientado a objetos se piensan en la resolución de problemas haciendo preguntas ¿Quién lo va hacer?, ¿Sobre quién se va actuar?, el utilizar la palabra "quién", me refiero a los objetos, ya que son estos las estrellas principales de este modelado. En la programación orientada a objetos se les visualiza como entidades que poseen una cierta "conducta", esto quiere decir que saben las tareas que hay que hacer y cómo hacerlas, es por eso que empezaremos a profundizar sobre el modelado orientado a objetos definiéndolos, existen varias definiciones, veamos algunas de ellas (ver Fig 1.7)

[4]Objeto. Es cualquier cosa, persona, entidad sobre la cual se desea almacenar información.

[5]Objeto. Es una entidad que tiene una función específica de la cual es responsable, y la información necesaria para cumplir con esa tarea. Un objeto posee estados, conducta e identidad.

[6]Objeto. Encapsulación genérica de datos y de los métodos para manipularlos, es una entidad que tiene unos atributos particulares, las propiedades, y unas formas de operar sobre ellos, los métodos.

Figura 1.7
Definición de Objeto

Como podemos ver todas las definiciones nos dicen que:

> un objeto es una entidad, que posee "conocimientos" para cumplir tareas específicas, y sobre el cual se va almacenar información en base a sus atributos.

Figura 1.8
Diagrama de Clases

[4] *ADORACIÓN DE MIGUEL Y MARIO PIATTINI en su Libro "Fundamentos y Modelos de Bases de Datos "*

[5] *ING ERNESTO PEÑALOZA ROMERO en su Libro "Fundamentos de programación"*

[6] *FRANCISCO JAVIER CEVALLOS en su libro "Java 2 Curso de Programación"*

Resumiendo, un objeto contiene Información (conocimiento), Funciones (conducta), Mensajes (entradas / salidas), Encapsulamiento (caja negra), y entre sus atributos nos encontramos con información oculta es decir privada, solo para el objeto, información pública, la cual puede ser vista por objetos externos y con información protegida, la cual puede ser vista por aquellas clases que se encuentran directamente relacionadas con la clase en cuestión, por lo tanto un objeto lo podemos ver de la siguiente manera: (ver Fig 1.9)

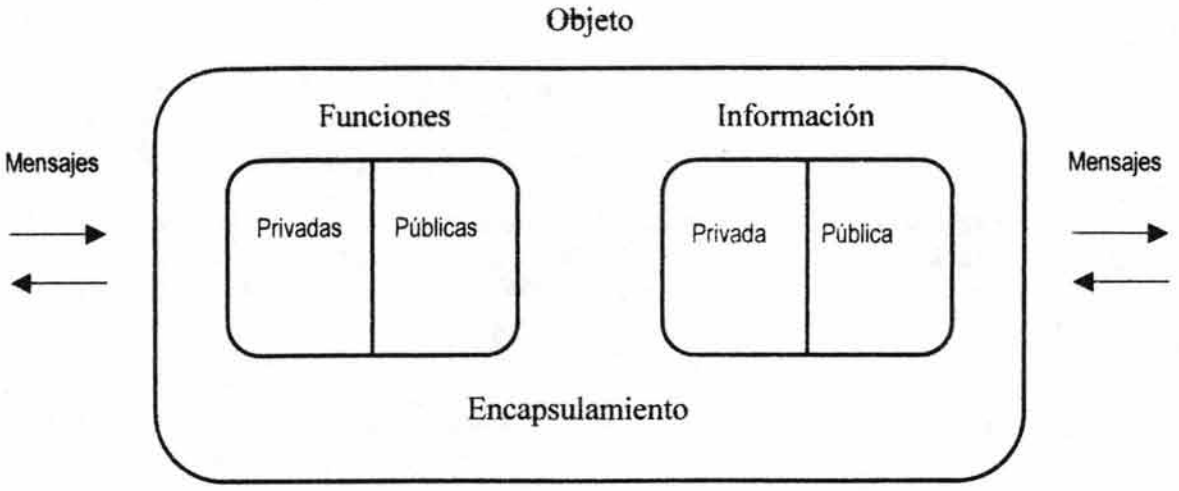


Figura 1.9
Definición de Objeto

Clase

Una clase se refiere al conjunto de n objetos, los cuales tienen características en común, estas características están contenidas en la propia clase, de tal manera que cada elemento toma las que le conviene, veamos la siguiente figura para entenderlo.(ver Fig 1.10)

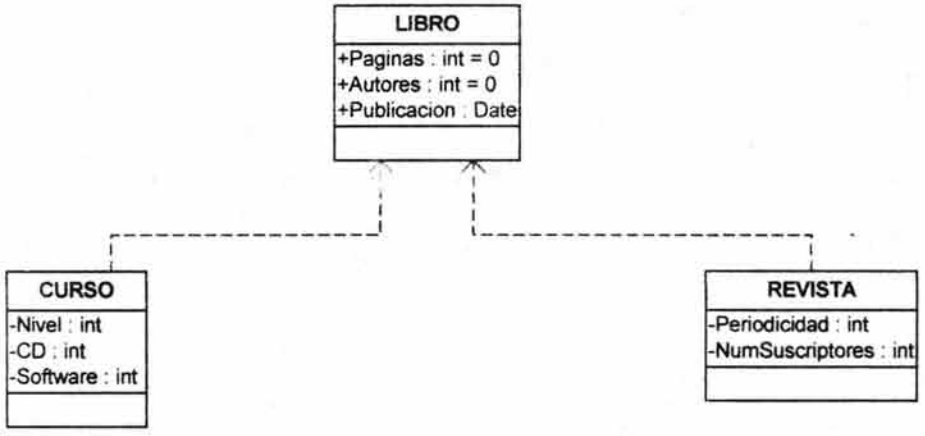


Figura 1.10
Diagrama de Clases

En el diagrama de clases expuesto anteriormente, se puede visualizar la clase libro, la cual tiene las características de páginas, autores y la fecha de la publicación, estas características son llamadas atributos de la clase LIBRO, dentro de la clase libro se encuentran subclases que según lo expuesto son; CURSO y REVISTA, estas clases contienen sus propios atributos, con la flecha hacia la clase libro nos indica que existe una relación con ella, es decir los atributos públicos de la clase LIBRO, son vistos por las clases de abajo. Para entenderlo mejor, la clase LIBRO hereda sus características a los objetos, miembros de la clase REVISTA, y de la clase CURSO, estos tienen características en común sin embargo cuentan con sus propias características, de esta manera podemos categorizar a los objetos en una jerarquía de clases, como ya he mencionado este concepto nos sirve tanto para el diseño del almacenamiento de información, como para la programación. Ahora bien enfocándonos a la programación y retomando que un objeto contiene funciones e información, veamos la funcionalidad:

En el diagrama anterior existe la clase "REVISTA", esta puede contener un conjunto de objetos[7], los cuales pueden ser REVISTA deportes, REVISTA entretenimiento, las cuales contendrán, las características públicas y privadas de la clase REVISTA, y también contendrán las características públicas de la clase "LIBRO", es decir, la clase REVISTA **hereda** las características públicas y protegidas de la clase superior, también llamada **superclase**, así es como se da la reutilización de código sin tenerlo que escribir otra vez, cuando se requiera un método o un atributo que ya existe en alguna de las clases, basta heredar para obtenerlos, por lo tanto la **herencia** nos viene a cambiar todo un concepto de programación, reduciendo así código y facilitando el diseño a través de la abstracción.

Por lo tanto, a partir de una clase definida con atributos y métodos propios, se crean objetos miembros de dicha clase, los cuales, cada uno de ellos cuenta con los métodos y atributos definidos en la clase. Es posible crear n objetos de una clase.

Paso por valor y paso por referencia

Una vez que ya entendimos los conceptos de clase y objetos, así como sus interrelaciones, existen atributos que pueden pasar de una clase a otra ya sea por valor o por referencia, cuando una clase hereda atributos por valor, estos tienen vida mientras los objetos de la clase superior existan, una vez que estos son destruidos, los atributos dejan de existir, en cambio si estos son heredados por referencia seguirán existentes ya que son independientes de los objetos de la clase superior.

Clase abstracta

Es aquella de la cual no se crearán o instanciarán objetos, es decir, la única manera de acceder a sus métodos abstractos es por medio de la instanciación de sus clases inferiores.

Mensajes

Los mensaje fluyen entre objetos y cada uno es encargado de procesarlos o bien canalizarlos hacia otros objetos.

[7] Llamados también instancias de la clase

Encapsulamiento

Como se ha estado mencionado, cada objeto es responsable de cumplir las funciones que a él fueron asignadas, ahora bien, el Encapsulamiento radica en que cada objeto posee sus propias funciones e información, las cuales no pueden ser visualizadas por otros objetos, de tal manera que no sea alterada su información si no es autorizado por el mismo, esto es realizado desde la definición del objeto,

Polimorfismo

No es mas que el comportamiento diferente de cada objeto a un mismo mensaje del exterior, por ejemplo, tenemos tres clases; INSERCIÓN, ACTUALIZACIÓN, CONSULTA que heredan de la clase TRANSACCIÓN_SQL, estas tres clases reciben el mismo mensaje; "Acceder a la Base de Datos", cada objeto miembro de cada clase va a recibir el mismo mensaje pero va a tener diferente comportamiento, como el nombre de cada clase lo dice; la clase INSERCIÓN realizará un acceso a la base de datos realizando una inserción de información, la clase ACTUALIZACIÓN realizará una acceso a la base de datos realizando una actualización de información y por último, la clase CONSULTA hará una extracción de información de la base de datos. Ver Fig. 1.11.

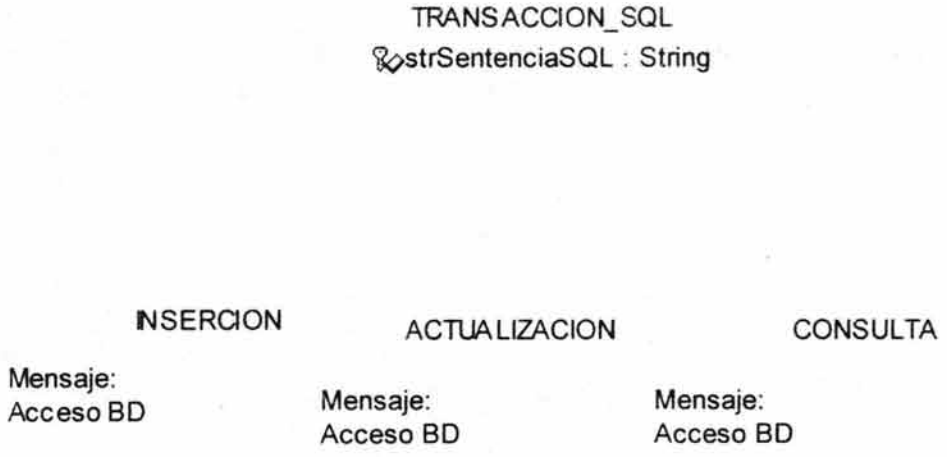


Figura 1.11
Ejemplo de polimorfismo

1.1.4.4 UML(Unified Modeling Language)

UML nos ofrece una forma de visualizar el flujo de información, los procesos del negocio y las funciones del sistema, así como en el diseño en caso que estuviéramos pensando en una jerarquía de clases, aquí es dónde se puede especificar los atributos de cada una de ellas, y conceptualizar todos aquellos elementos existentes en el sistema, al mismo tiempo lo estamos documentando, al igual que el diagrama de flujo, la visualización de los procesos, la conceptualización de las funciones deben ser totalmente entendibles para un analista ajeno al desarrollo del sistema, con esto podemos evitar las confusiones en el momento de dar mantenimiento al mismo o bien agregar módulos.

Al utilizar la metodología UML, nos encontramos con el planteamiento de clases, sus relaciones y sus atributos, tal y como lo vimos en el apartado anterior, en esta metodología podemos visualizar las relaciones específicas entre clases.

En UML, una clase es representada por un rectángulo que posee tres divisiones. Esto lo podemos hacer en [8]Rational Rose. (ver Fig 1.12)

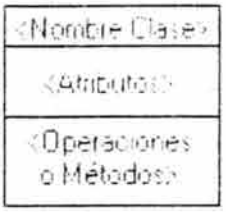


Figura 1.12
Clase en UML utilizando Rational Rose

Ahora veamos un ejemplo de clase llamada "CUENTA", con algunos atributos, los cuales tienen un alcance. Ver Fig 1.13

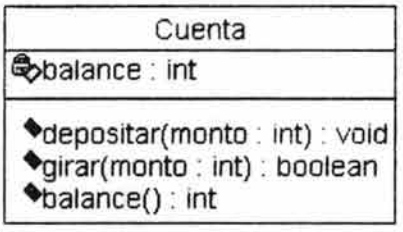


Figura 1.13
Ejemplo de clase con atributos

Como vimos en el apartado anterior, los atributos y métodos de la clase tienen diferente alcance, pueden ser público, privados o protegidos, en Rational Rose estos alcances se indican con diferentes figuras, así como los métodos y las relaciones con otras clases, ver Fig. 1.14,

[8] Software utilizado para el diseño de sistemas, el cual facilita el desarrollo de los mismos utilizando sus herramientas como ingeniería inversa.

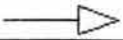
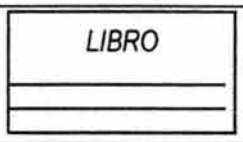
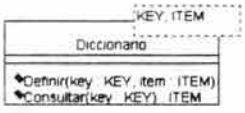
Símbolo	Descripción
	Atributo público
	Atributo privado
	Atributo protegido
	Método público
	Método Privado
	Método Protegido
	Herencia
	Paso por valor
	Paso por Referencia
	Asociación
	Dependencia o Instanciación
	Clase Abstracta
	Clase parametrizada

Figura 1.14
Simbolización en Rational Rose

Algunos de los símbolos expuestos en la figura 1.14 ya los hemos explicado en el apartado anterior, veamos los demás;.

Asociación.- Es una relación entre clases que indica que sus objetos colaborarán juntos sin tener una dependencia entre sí.

Dependencia o instanciación.- relación que nos indica que los objetos de la clase inferior serán totalmente dependientes de su superclase.

Clase parametrizada.- Es aquella clase que para ser instanciada necesita de ciertos parámetros.

Casos de uso

Los casos de uso se utilizan para representar la interacción que sucederá entre los actores y el sistema, son documentos narrativos que describen la secuencia de eventos de un actor que utiliza un sistema para terminar un proceso. Definamos los elementos que componen a un caso de uso:

- Actor.- es la relación que tiene un usuario con el sistema, este no necesariamente es una persona, también puede ser un subsistema.
- Caso de uso.- Es la tarea que se realiza tras la petición, orden de un actor u otro caso de uso.
- Relación.- las relaciones existentes son las que se expusieron en la figura 1.14.

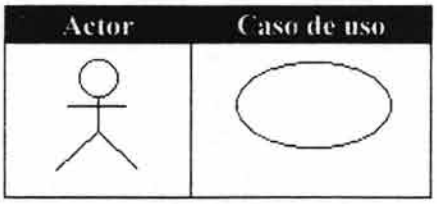


Figura 1.15
Representación en
Rational Rose

Ejemplo. El actor es el auxiliar de personal, el cual va ordenar al sistema la generación de la nómina, y este último es el caso de uso, su representación la vemos en la fig 1.16

Figura 1.16
Ejemplo Caso de uso

Diagrama de interacción

En un diagrama de interacción, representamos los objetos que interactúan entre sí, una vez que sucedió un evento. Sus elementos se encuentran en la fig 1.17.

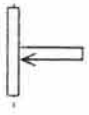
Elemento	Descripción
	Actor (Objeto)
	Mensaje a otro objeto
	Mensaje al mismo objeto

Figura 1.17
Ejemplo Elementos de
Un Diagrama de interrelación

1.1.4.5 Diseño y Desarrollo de la Base de Datos

1.1.4.5.1. Modelo Entidad - Interrelación

Captar la semántica del mundo real y lograr la interpretación del mismo en un modelo de datos es el objetivo del diseño de la BD.

Cuando nos referimos a una semántica de datos, estamos hablando de las reglas de negocio, que no son mas que reglas establecidas y generadas de acuerdo a las necesidades del mismo, en un principio esta semántica de datos solo se encontraba en la mente del usuario, con el tiempo se plasmó en los sistemas hechos en algún lenguaje de programación, posteriormente nos encontramos que parte de esta semántica la podemos plasmar en la misma BD, esto nos repercute directamente en el diseño, ya que desde aquí podemos cubrir ya algunos requerimientos para resolver problemas.

El modelo de datos es con el que debe de comenzar un analista para diseñar su BD, antes de montar esta en un Sistema Gestor de Base de Datos (SGBD), el modelado de datos se realiza independiente del SGBD, ya que primero se debe de tener una visión conceptual de los datos que contendrá nuestra BD, investigar dentro del entorno del negocio, los datos que se encuentran ya definidos en formatos, reportes, archivos de texto, etc, y además investigar aquellos que no se encuentran definidos y que los usuarios dan por hecho de que existen. Sin embargo esta información solo se encuentra en la mente del usuario, y es nuestra misión obtenerla, interpretarla y conceptualizarla, y así obtener nuestro Diagrama Conceptual, dónde plasmaremos todos estos datos obtenidos del negocio con la descripción de cada uno de ellos, así como en gran medida sus interrelaciones. Sucede mucho que el analista se salta esta etapa del diseño de la BD, y es que le resulta más sencillo interpretar solamente la estructura percibida en las entrevistas e investigación de requerimientos y realiza directamente el esquema de la BD, esto sucede con frecuencia, y por lo tanto se obtiene un mal diseño y como consecuencia, cuando la BD se encuentre en producción, ésta sufrirá cambios físicos que afectarán la lógica, reglas, incluso la programación de las aplicaciones que tengan acceso a ella.

Adoración de Miguel Mario Piattini en su libro " Fundamentos y Modelos de Bases de datos ", nos menciona las características de los modelos conceptuales. Ver fig 1.18.

- ✓ No son implementados en SGBD
- ✓ Son independientes del SGBD
- ✓ Tienen mayor nivel de abstracción
- ✓ Tienen mayor capacidad semántica

Figura 1.18
Características del

Modelo Conceptual

Este es un modelo de datos semánticos, el cual nos ayudará a diseñar nuestra BD desde un punto conceptual, de esta manera pensamos solamente en la concepción de los datos y por ahora, no nos preocuparemos de la implantación en un SGBD. Este modelo fue propuesto por Peter P. Chen (1976).

Los conceptos en los cuales este modelo se basa son:

Entidad

Estamos hablando de un objeto (explicado en la sección 1.4.3) sobre el cual necesitamos almacenar información, es decir las ocurrencias de la entidad. Existen dos tipos de entidades; las regulares, esto es, las ocurrencias de esta entidad existen por sí mismas, y las débiles, mientras que las ocurrencias de esta entidad existen mientras existan las ocurrencias de una entidad regular a la cual pertenece, cada una tiene su forma de representación en el diagrama. En el ejemplo de la Fig. 1.19, las ocurrencias de la entidad débil EMPLEADO contienen cada uno un puesto, por lo tanto, EMPLEADO depende de puesto, ya que si se elimina una ocurrencia de PUESTO, se tendrá que modificar forzosamente a los empleados que tenían ese puesto, ya que un empleado no puede tener un puesto que no existe.

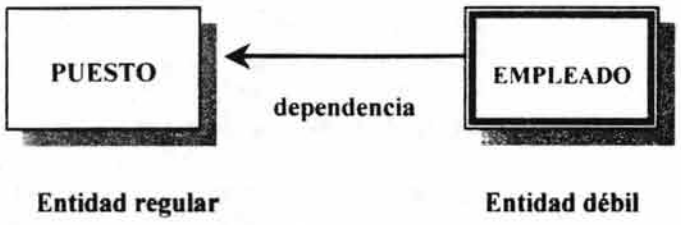


Figura 1.19
Tipos de entidades

Interrelación

Esta es una asociación entre entidades, como ya lo mencioné, las entidades son objetos, y como se expuso en la sección 1.4.4, los objetos tienen relaciones entre sí, así que ahora lo veremos conceptualmente con el modelo entidad – interrelación, pongamos el ejemplo de un negocio de desarrollo de software, tenemos dos entidades, ANALISTA y SISTEMA, la interrelación entre estas dos entidades es " Documenta " como se expone en la Fig. 1.20.

Figura 1.20
Interrelación de
Entidades

Atributo

Como vimos en la sección " Clases y Objetos ", un atributo es una característica de la entidad, objeto o clase, en este caso es igual, en la Fig 1.21 los atributos de la entidad " SISTEMA ", pueden ser; Fecha_de_liberación, Plataforma, Requerimientos, etc, así como también existen tipos de atributos, regresémonos un poco al ejemplo de la Fig 1.20, aquí se plantea una dependencia entre entidades, y se habla que las ocurrencias de una existen ciertamente si existen en la entidad regular, entonces también habíamos mencionado que ocurrencias son los diferentes valores que puede tomar un campo de datos, y esto ultimo vendría siendo sin mas preámbulos un atributo. El conjunto de valores que puede tomar un atributo se llama dominio, por ejemplo, asignando un atributo a la entidad SISTEMA llamado " Plataforma ", su dominio es " plataformas ", cuyos valores pueden ser: ' UNÍX, Windows NT, AS400 ', por lo tanto el atributo " Plataforma " tomará los valores de su dominio. Es claro que una entidad puede contener varios atributos, uno de ellos debe ser " indentificador principal " que será el identificador de cada una de las ocurrencias, esto dicho en otras palabras, en este atributo es forzoso que contenga valor, los demás pueden ser alternativos, sin embargo como estamos en nuestro modelo conceptual, pueden existir atributos candidatos a ser principales. Para nuestro modelo la representación de lo anterior lo podemos ver en la Fig. 1.21.

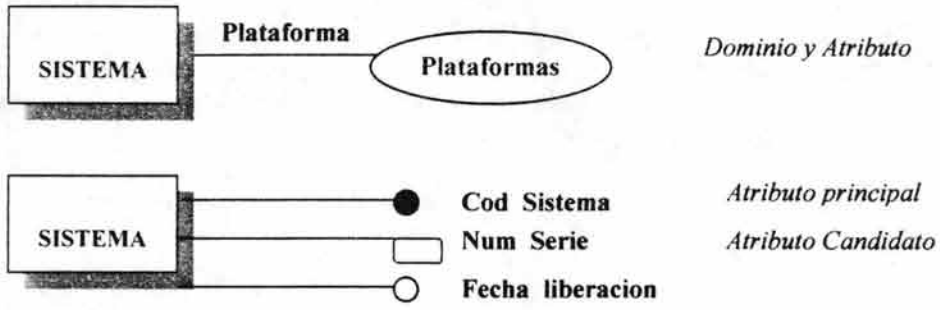


Figura 1.21
Representación de
Dominio y atributos

Cardinalidad de entidades

Son el número máximo y mínimo de ocurrencias de una entidad que pueden estar relacionadas con otra. Así por ejemplo, en la Fig. 1.22 la entidad " SISTEMA " tiene una Cardinalidad de (1,n), esto quiere decir que un " ANALISTA " puede " Documentar " 1 o n " SISTEMAS ", y la entidad " ANALISTA " tiene como Cardinalidad (1,n), es decir un " SISTEMA " puede ser documentado mínimo por 1 " ANALISTA " o por muchos. Ver Fig. 1.22.

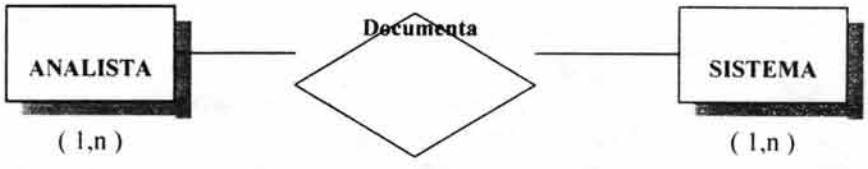


Figura 1.22
Cardinalidad de entidades

Hasta este momento tenemos el modelo conceptual, en el siguiente apartado veremos el diagrama de la BD en base al modelo conceptual planteado.

1.1.4.5.2. Bases de Datos

Una Base de Datos es un modelo del mundo real, aquí se plasman todos los datos que de alguna u otra manera son utilizados diariamente por las personas, estos datos están detenidamente interrelacionados por medio de métodos bien definidos, de esta manera se crea una concordancia entre ellos, así como una consistencia e integridad de la información, la cual se vuelve un objetivo de la Base de Datos, es decir, esta se encargará de almacenar toda esa información encargándose también de gestionar a los usuarios que la van utilizar, otorgando permisos .Cuándo se realiza un análisis tenemos que identificar todos los objetos existentes en la atmósfera del negocio, organización, institución para el cual estaremos haciendo el sistema.

En una BD(utilizaré esta abreviatura para referirme a Base de Datos) existen varios objetos como propios de ella, estos son Tablas, Relaciones, procedimientos, etc, conforme se está realizando el diseño, también se van generando estos objetos, cuidando los objetivos y las operaciones que ocurrirán en ella una vez que está en producción, hablaré un poco de ellos antes de entrar a lo que es en sí el diseño de la BD de nuestro sistema.

Una vez que se tiene el modelo conceptual, realizamos el esquema de la BD en base a él, acercándonos cada vez mas al SGBD, nos podemos auxiliar de herramientas que nos facilitan la esquematización de la BD y la implementación en el SGBD, en el capítulo IV el diseño de la BD de nuestro sistema lo haré utilizando Power Designer, y en el siguiente apartado explicaré el Modelo entidad – relación, posteriormente el diseño realizado se monta en el SGBD, y aquí es dónde realizamos un esquema interno de acuerdo a las características del SGBD, por ultimo se llena la BD con las ocurrencias[9] del negocio. Ver fig 1.23.

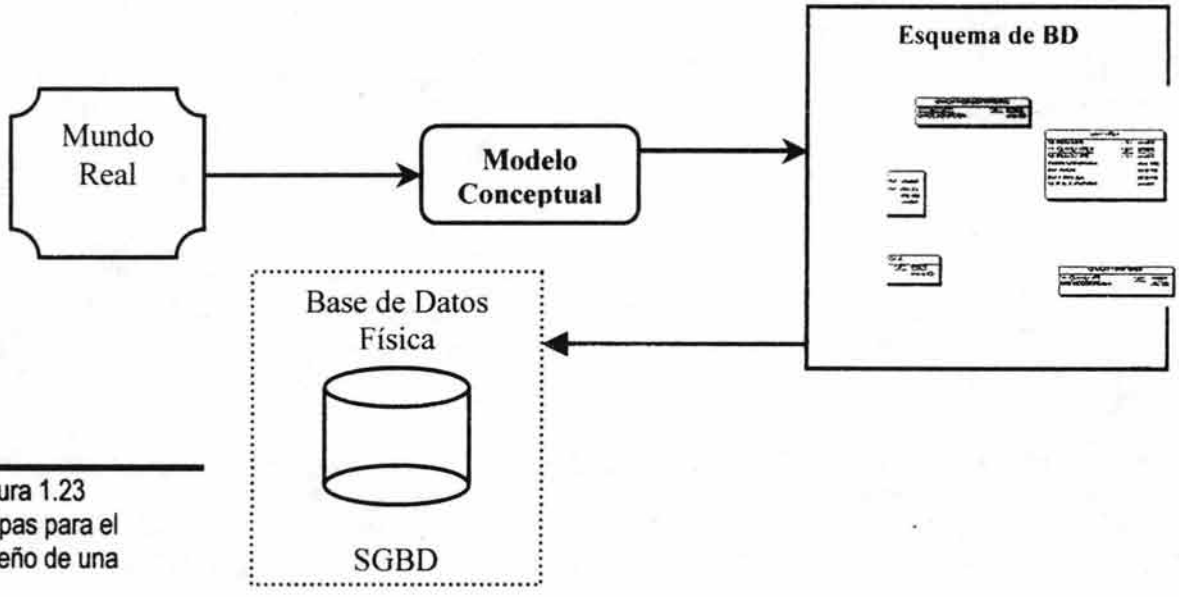


Figura 1.23
Etapas para el
Diseño de una

Tabla.- Es un objeto de la BD que va albergar cierta información para la cual fue diseñada, esta contiene un número definido de campos y ciertas propiedades, como las relaciones con las demás tablas por medio de sus campos llave, y los permisos que existen sobre ella en cuanto a acceso y movimientos. La información se almacena en registros, el número de registros que pueda almacenar esta, depende del Gestor de Base de datos que se esté utilizando.

[9] Son los valores que tomarán todos los campos de las tablas de la BD.

Campo.- También es llamado atributo, es una columna de datos, definida de un tipo de dato: Integer, Decimal, String, etc. Existen diferentes tipos de campos:

Llaves .- Es un concepto que se usa para identificar a un campo a un conjunto de ellos, cuándo esto se hace de forma única se dice que es una **llave primaria (primary key)**, de lo contrario estaremos hablando de una **llave secundaria (foreing key)**, esta se utiliza para identificar a un conjunto de registros pertenecientes a un registro en otra entidad por medio de lo o los campos llave. Estas llaves sirven como base para las **relaciones entre tablas**, es decir de una entidad – relación.

Índice.- Es el ordenamiento lógico de los registros, por medio de índices el acceso a la información es más rápido porque estos utilizan un método de búsqueda, es por eso que ordenan la información por los campos especificados.

Registro.- Es un conjunto de datos que tienen relación con la entidad (tabla) en la que se encuentra, es decir, es en sí la información organizada en filas, las cuales respetan las características de la tabla o entidad.

Redundancia.- es la duplicidad en la información, en las BD nos encontramos con lo que se llama redundancia controlada, muchas veces por eficiencia es necesario permitir una redundancia en la información, siempre y cuando el sistema se encargue de controlarla, realizando absolutamente todas las actualizaciones pertinentes en todos los lugares dónde sea necesario, aunque el usuario no verá esta redundancia física, ya que no debe de existir una redundancia lógica.

Integridad.- la información contenida en la BD debe ser coherente y consistente de acuerdo a las reglas semánticas propias del negocio.

Seguridad.- la información debe estar protegida contra deterioros por causas físicas y lógicas, así como también a accesos no autorizados.

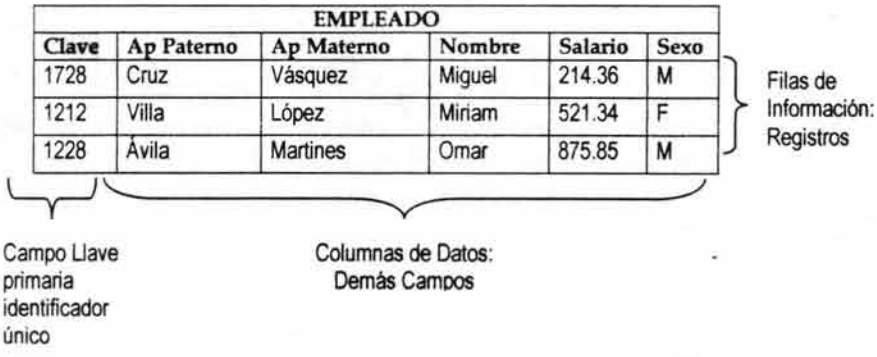


Figura 1.24
Elementos de una tabla
De una Base de Datos

1.1.4.5.3. Desarrollo de la Base de Datos (Modelo Relacional)

Ahora nos acercamos más a la implantación de la BD en el SGBD, interpretando el modelo de datos conceptual visto en el apartado anterior, armaremos el diagrama de base de datos, para esto necesitamos traducir los elementos que hemos visto desde el modelo orientado a objetos hasta el modelo conceptual de datos. Ver Fig 1.25

Objeto	En una BD un objeto puede ser una Tabla, Relación o Procedimiento.
Atributo o Característica	Es la columna o campo de una tabla, un atributo principal viene siendo una " Llave primaria ", un atributo candidato es una llave foránea.
Tipo de dato	Numérico, carácter, cadena de caracteres, fecha, etc.
Ocurrencias o Valores	Son filas de información (registros) almacenadas en cada tabla.
Interrelación	Referencia de una tabla a otra.

Figura 1.25
Traducción de conceptos
a las Bases de Datos

Retomemos el ejemplo de la figura 1.22, para realizar su BD física, utilizaré la herramienta [10]Power Designer para el diagrama, eligiendo como SGBD SQL Server 6.x según nuestro modelo de datos conceptual, nos indica la existencia de dos entidades, ANALISTA y SISTEMA, así como una interrelación llamada Documenta, cada entidad tiene sus propios atributos, así que los especificaremos traduciendo aquellos atributos principales como llaves primarias (Primary Key), y los atributos candidatos como llaves foráneas (foreing key), para estas últimas se respetará el concepto de dependencia visto en el apartado anterior, es decir, la dependencia de una tabla a otra, los registros de la tabla dependiente existirán siempre y cuando existan los registros de la tabla independiente. En este caso en la tabla ANALISTA, podemos observar que nuestra llave primaria <pk> es INTEMPLEADO, es decir la clave del empleado, este es un identificador único para el mismo, por lo tanto no es posible que dos analistas tengan el mismo identificador de empleado, que es una de las características de una llave primaria, así como no puede estar vacío este campo, esto lo denotamos con un "not null" (no permite nulos), el tipo de dato es "numero entero" int, y los demás campos son varchar(60), es decir del tipo cadena variable de longitud máxima de 60 caracteres y si permiten nulos. El nombre de los campos tiene un prefijo para identificar rápidamente de qué tipo de datos es: INT – entero, CHR – carácter o cadena de caracteres, DAT - fecha. En el mismo Power Designer podemos documentar nuestra BD conforme la vamos creando, asignando descripciones de las tablas, incluso de cada campo, claro esto lo podemos hacer, sin embargo, por lo general existen formatos de documentación ya establecidos por cada negocio.

[10] Herramienta CASE, para diseñar dragramas físicos de Base de Datos, especificando el Sistema Gestor de Base de Datos, esto lo veremos en el capítulo V

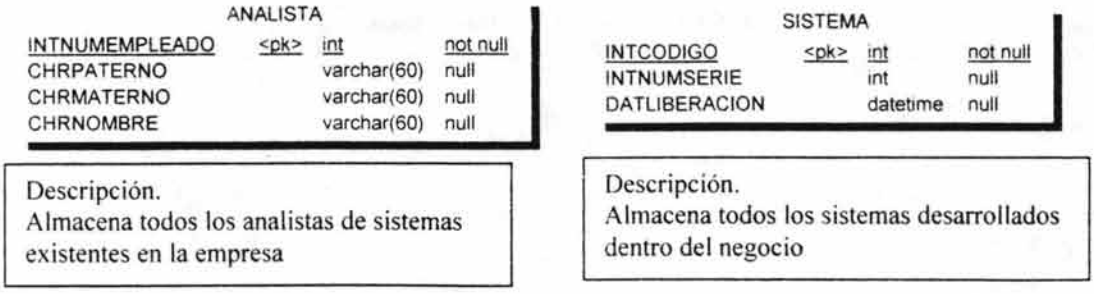


Figura 1.26
Tablas realizadas
en Power Designer

Una vez que ya tenemos nuestras tablas, tenemos que relacionarlas con la interrelación Documenta de nuestro modelo conceptual, ver Fig. 1.27, esta relación se denota con una flecha que parte desde la tabla ANALISTA a la tabla SISTEMA, esto quiere decir que en el momento que nosotros dibujemos en Power Designer una relación, automáticamente se copiará el campo que es llave primaria de la tabla SISTEMA, a la tabla ANALISTA, como llave foránea <fk>, con esto indicamos que un ANALISTA no puede documentar sistemas que no se encuentran registrados en la BD, también podemos ver la Cardinalidad establecida en el modelo conceptual y los índices establecidos para cada tabla llamados EMPLEADO y SISTEMA.

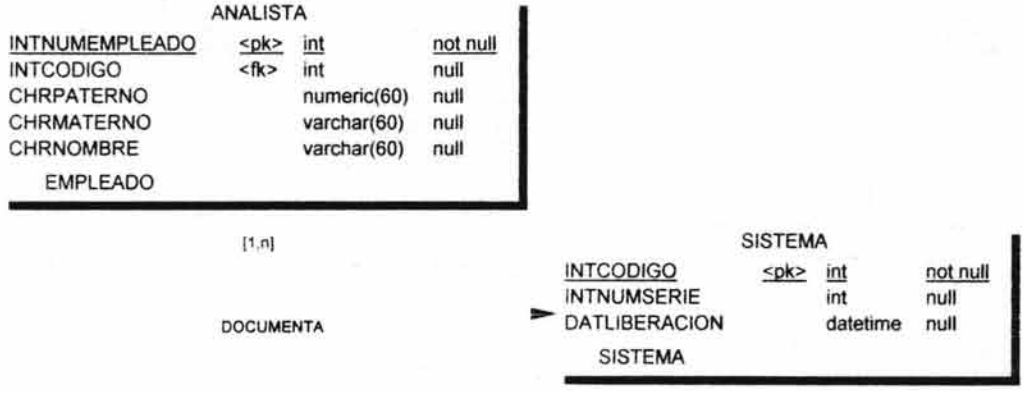


Figura 1.27
Relación entre
ANALISTA y SISTEMA

Veamos ahora las sentencias SQL que crean las tablas en el SGBD, Power designer nos da la facilidad de generar estas sentencias una vez capturadas las tablas y sus relaciones. El Script lo podemos ver en la Fig. 1.28, este se correrá en el editor SQL de SQL Server 2000, y obtenemos el diagrama de la BD ya creada en ese SGBD. Ver Fig. 1.29.

Tabla ANALISTA

```
create table ANALISTA
(
   INTNUMEMPLEADO   int            not null,
   INTCODIGO        int            null   ,
   CHRPATERNO       numeric(60)    null   ,
   CHRMATERNO       varchar(60)    null   ,
   CHRNOMBRE        varchar(60)    null   ,
   constraint PK_ANALISTA primary key
(INTNUMEMPLEADO)
)
```

Tabla SISTEMA

```
create table SISTEMA
(
   INTCODIGO        int            not null,
   INTNUMSERIE      int            null   ,
   DATLIBERACION    datetime       null   ,
   constraint PK_SISTEMA primary key (INTCODIGO)
)
```

Relación DOCUMENTA

```
alter table ANALISTA
   add constraint FK_ANALISTA_DOCUMENTA_SISTEMA foreign key (INTCODIGO)
     references SISTEMA (INTCODIGO)
```

ÍNDICES

```
create index SISTEMA on SISTEMA (DATLIBERACION, INTNUMSERIE)
go
create index EMPLEADO on ANALISTA (CHRPATERNO, CHRMATERNO, CHRNOMBRE)
go
```

Figura 1.28
Script de la BD

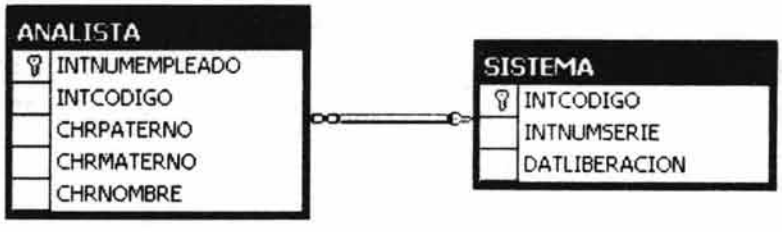


Figura 1.29
Diagrama de la BD
en SQL Server 2000

1.1.5 Desarrollo de un Sistema de Información

En el apartado de Análisis y diseño de un sistema de información, hablé del ciclo de vida de un sistema, del tiempo de diseño, ahora nos toca ver el desarrollo del mismo, comenzaré por plantear un panorama general de un Sistema de información, omitiendo los conceptos ya explicados en el apartado mencionado.

En el presente apartado veremos de una forma más técnica y concreta el desarrollo de un Sistema de Información, dónde explicaré de una forma general las herramientas que se utilizan para almacenar, gestionar y manipular la información. Pero antes veremos la lógica general del funcionamiento de un sistema de información.

Utilizando la arquitectura cliente – servidor nuestro sistema funcionará de la siguiente manera:

Todos los clientes tendrán acceso a él, por medio de un Navegador de Internet, en ellos se ejecutará la etapa dedicada a la interfaz de usuario y en el servidor Web se ejecutarán todos los procesos necesarios. En el diagrama de la Figura 1.30 podemos apreciar la arquitectura básica para que trabajemos con nuestro sistema de información.

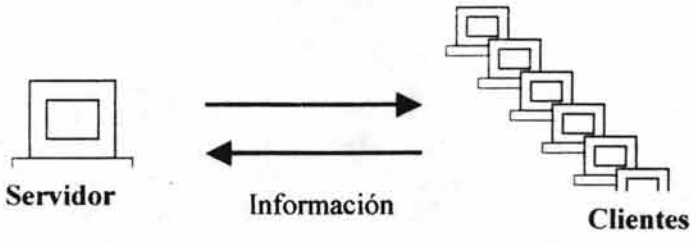


Figura 1.30
Cliente Servidor

Cuando surgieron las redes computacionales, surgió también la necesidad de realizar sistemas que se interconectaran entre sí, logrando una comunicación plena, esto fue conocido como cómputo distribuido. Empezaron a surgir sistemas que estaban desarrollados en una sola capa, es decir, la interfaz, los procesos y el acceso a la información se encontraban en el mismo programa, y por lo tanto cada uno de estos elementos eran dependientes entre sí. Como se muestra en la Fig. 1.31A

Figura 1.31A
Desarrollo en una capa

Posteriormente la creación de nuevas metodologías dieron el surgimiento del desarrollo a dos capas, este implica que la Interfaz de usuario se encuentra en el cliente, esta se comunica para realizar cualquier proceso al servidor, los procesos, es decir las reglas de negocio y la información se encuentran en una base de datos, asi que el cliente realiza peticiones al servidor, este ultimo ejecuta lo requerido en la propia base de datos que es la misma que proporciona la información, y regresa una respuesta al cliente. Los procesos se almacenan en

procedimientos almacenados. Con este desarrollo podíamos hablar ya de una arquitectura cliente – servidor. En este tipo de desarrollo la interfaz es independiente de las reglas de negocio, veámoslo mas claramente en la Fig. 1.31 B.

Figura 1.31 B
Arquitectura en 2 capas

Mas adelante, se pensó en realizar una separación mas, es decir un desarrollo a tres capas, este nos permite tener la Interfaz gráfica de usuario independiente de las reglas de negocio, estas a su vez independiente de la Base de datos, con este nuevo desarrollo podemos pensar en la portabilidad de cada uno de los elementos, para lograr esto, podemos pensar en las tecnologías del desarrollo orientado a objeto, que nos permiten un diseño de reutilización. Y con esto podemos hablar de una estructura cliente – servidor mejorada. Estamos hablando de un diseño arquitectónico que nos permitirá soportar los cambios que se requieran en el software de una manera más sencilla y eficiente. Veámoslo en la Fig. 1.31 C

Figura 1.31 C
Arquitectura en 3 capas

Como hemos observado, el diseño arquitectónico de los sistemas de información tiende a dividir cada vez mas en módulos la operación de un sistema. Hoy día se realizan sistemas en 'n' capas, así es, se realizan más divisiones utilizando las nuevas tecnologías tanto de programación como de Base de datos. Veamos el diseño

arquitectónico de cuatro capas, este consiste en separar el módulo de acceso a datos de las reglas de negocio, esto significa mayor eficiencia en nuestro sistema, ya que las modificaciones posteriores no deben ser tan complicadas, es más el módulo que ahora es independiente (acceso a datos) puede ser flexible y aceptar diferentes orígenes de datos, esto sin modificar las reglas de negocio. Veamos la Fig. 1.31 D.

Figura 1.31 D
Arquitectura en 4 capas

1.1.5.1 Sistema Gestor de Bases de Datos

Un Sistema Gestor de Base de Datos (SGBD) es aquel que administra todo lo concerniente a una o varias bases de datos, en él se encuentran todos los objetos miembro de la BD, como:

- **Bases de Datos**
  - **Tablas**
    - **Campos**
  - **Diagrama**
  - **Vistas**
  - **Procedimientos Almacenados**
  - **Usuarios**
  - **Funciones**
  - **Reglas**
  - **Tipos de datos definidos por el usuario**
- **Seguridad**
  - **Inicios de sesión**
  - **Permisos**
- **Copias de seguridad**
- **Monitoreo de actividades**
- **Planes de mantenimiento a la base de datos**
- **Transacciones con otros servidores de Bases de Datos**
- **Importar y exportar información, meta datos de otros orígenes de datos**

Cada SGBD nos da las ampliaciones o limitaciones, es por eso que cuando desarrollamos un Sistema de Información, hay que estudiar los diferentes SGBD existentes en el mercado, cuál va a cubrir nuestras necesidades al 100%. En esta decisión tenemos que recordar que el SGBD puede ser muy bueno, incluso superar nuestras necesidades, cuidado con esto ya que no siempre es bueno, en un sistema que no planea grandes cantidades de información sería un gasto mas que una inversión el SGBD.

El SGBD nos proporciona una interfaz adecuada para que el usuario manipule tanto los datos definidos, como la información contenida en la BD, también nos da la facilidad de establecer conexión con otros orígenes de datos, de tal manera que podemos realizar una transferencia de información de un origen de datos a otro.

Como vimos en el listado anterior, el SGBD contiene a la base de datos, por lo tanto, la lógica de comunicación con las aplicaciones que requieran información de una BD, antes tienen que entenderse con del SGBD, el cual les va a proporcionar los servicios de acceso a la base de datos requerida. Es decir, un cliente requiere realizar una conexión a una Base de datos llamada RECURSOS_HUMANOS, contenida en el SGBD SQL Server que reside en un servidor llamado DSCEAD_SERVER conectado a una LAN[11], por lo tanto por medio de un controlador se realiza la comunicación, el SGBD verificará:

- ✓ **Si existe la Base de datos**
- ✓ **Si el usuario que solicitó la conexión tiene acceso a ella**
- ✓ **Y posteriormente si el usuario tiene permisos para las acciones que quiera ejecutar como consultas de información, nuevas definiciones de datos, etc.**

Otra de las facilidades que nos proporciona un SGBD es un lenguaje de definición de datos y de manipulación de datos, con este podemos manipular la información desde uno de los programas del SGBD sin tener que realizar

[11] Local Area Network. Red de área local.

una conexión desde una de las aplicaciones que requieren de la Base de datos, este es el lenguaje SQL, del cual hablaremos a detalle en el siguiente apartado. Veamos en la Fig. 1.31E el diagrama que nos proporciona el libro Fundamentos y Modelos de Bases de Datos[12].

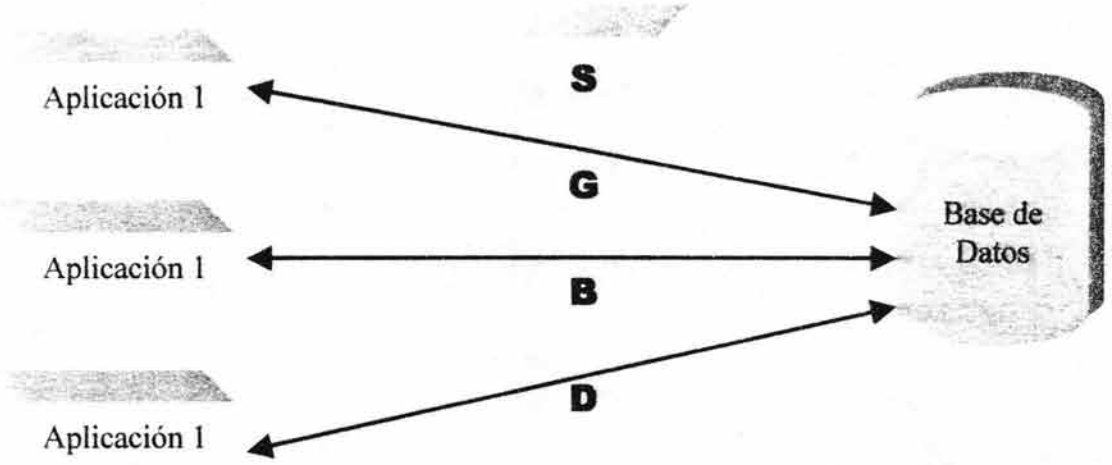


Figura 1.31E
Arquitectura en 3 capas

Por mencionar algunos SGBD:

- Microsoft SQL Server
- ORACLE
- Informix
- MySQL, etc.

[12] De Adoración de miguel, Mario Piattini.

1.1.5.2 Lenguaje SQL

Structured Query Language, el Lenguaje Estructurado de Consultas, es un lenguaje de definición y manipulación de datos, creado por IBM, posteriormente fue estandarizado en 1986 como el Lenguaje estándar para consultas a las bases de datos.

SQL como Lenguaje de Definición de Datos

Las instrucciones SQL en la definición de datos nos permite definir la estructura de toda una base de datos, desde la creación de la misma hasta la definición de todos sus objetos incluyendo relaciones y permisos. Expondré las instrucciones básicas en SQL para la definición de datos, antes aclaro que estas instrucciones pueden cambiar un poco dependiendo del SGBD que se esté utilizando.

- Para crear una Base de Datos llamada RECURSOS_HUMANOS

  create database RECURSOS_HUMANOS

  y para borrarla

  drop database [RECURSOS_HUMANOS]

- Para crear una tabla llamada RH_OPERANDO_FUENTE con 6 campos de diferentes tipos de datos e indicando cuáles son requeridos y cuáles no, por medio de la instrucción not null. También podemos observar que dos de los campos va ser llave primaria esto es con la instrucción **constraint** PK_RH_OPERANDO_FUENTE **primary key**. Esta tabla se crea en la BD activa

```
create table RH_OPERANDO_FUENTE
(
  SMI_COD_OP_FUENTE          smallint       not null,
  STR_COD_OP_FUENTE          varchar(10)    not null,
  STR_TABLA                  varchar(30)    not null,
  STR_CAMPO                  varchar(30)    not null,
  STR_DESCRIPCION            varchar(60)    not null,
  constraint PK_RH_OPERANDO_FUENTE primary
key(SMI_COD_OP_FUENTE,STR_COD_OP_FUENTE)
)
```

- Para Agregar un campo llamado CAMPO_A a la tabla RH_OPERANDO_FUENTE.

```
alter table RH_OPERANDO_FUENTE
ADD CAMPO_A SMALLINT NULL;
```

- Para borrar dicha tabla con todos sus campos y todos sus registros.

  drop table RH_ OPERANDO_FUENTE

- Para crear un índice LLAMADO IX_PERIODO a la tabla llamada RH_PERIODO, compuesto por el campo SMI_NUM_PERIODO

```
create index IX_PERIODO on RH_PERIODO (SMI_NUM_PERIODO)
```

- Para crear una relación llamada FK_PUESTO_PERTENECE_NIVEL entre dos tablas llamadas RH_PUESTO y RH_NIVEL, dónde la llave foránea en la tabla RH_PUESTO es el campo SMI_COD_NIVEL que al mismo tiempo viene siendo llave primaria para la tabla RH_NIVEL..

```
alter table RH_PUESTO
  add constraint FK_PUESTO_PERTENECE_NIVEL foreign key (SMI_COD_NIVEL)
    references RH_NIVEL (SMI_COD_NIVEL)
```

SQL como Lenguaje de Manipulación de Datos

Una vez que creamos la BD con sus tablas y a estas se le asignaron relaciones, subiremos información ayudándonos de las herramientas de cada gestor y así podremos utilizar SQL como Lenguaje de Manipulación de Datos.

- Para insertar un registro en una tabla llamada RH_EMPLEADO, respetando los tipos de datos contenidos en ella como también el orden

```
insert into RH_EMPLEADO values(1,1,'HERRERA','CASAS','EDGAR','08:30','17:30','2005-09-29','1978-01-
03','DISTRITO FEDERAL',1,NULL,'MEXICANO','M',1)
```

- Para buscar todos los campos del registro insertado.

```
Select * from RH-EMPLEADO where INT_COD_EMPLEADO = 1
```

- Para seleccionar solo algunos campos del registro insertado.

```
Select INT_COD_EMPLEADO, STR_PATERNO, STR_MATERNO, STR_NOMBRE from RH_EMPLEADO
where INT_COD_EMPLEADO = 1
```

- Para seleccionar varios campos de dos tablas previamente relacionadas, se necesita realizar un enlace (inner join on) .

```
select    RH_EMPLEADO.INT_COD_EMPLEADO, RH_EMPLEADO.STR_APPATERNO,
RH_EMPLEADO.STR_APMATERNO, RH_EMPLEADO.STR_NOMBRE,
            RH_DIRECCION_EMPLEADO.STR_CALLE, RH_DIRECCION_EMPLEADO.STR_COLONIA,
RH_DIRECCION_EMPLEADO.STR_DELEGACION
from      RH_DIRECCION_EMPLEADO inner join
            RH_EMPLEADO ON RH_DIRECCION_EMPLEADO.INT_COD_EMPLEADO =
RH_EMPLEADO.INT_COD_EMPLEADO
where     (RH_EMPLEADO.INT_COD_EMPLEADO = 1)
```

- Para borrar el registro insertado identificándolo por su clave principal.

```
delete from RH_EMPLEADO where INT_COD_EMPLEADO = 1
```

Como podemos observar, la cláusula SELECT nos sirve para seleccionar campos de diferentes tablas, por medio de criterios que se establecen en la instrucción WHERE, las tablas se especifican por medio de la instrucción FROM. La cláusula SELECT puede utilizar tantas funciones como contenga el gestor en el cual se está trabajando.

- Para obtener el número total de registros existentes en la tabla RH_EMPLEADO.

```
Select count(*) from RH_EMPLEADO
```

Función	Descripción
COUNT()	Devuelve la cantidad de registros encontrados
SUM()	Retorna la suma algebraica de un campo numérico
AVG()	Retorna el promedio de una columna numérica
MIN()	Retorna el mínimo del campo
MAX()	Retorna el máximo del campo
LOWER()	Retorna el texto en minúsculas
UPPER()	Retorna el texto en mayúsculas
LEFT(campo,n)	Retorna el numero de caracteres especificado de la izquierda
RIGHT(campo,n)	Retorna el numero de caracteres especificado de la iderecha

Por ultimo veremos las instrucciones de ordenamiento de la selección de información.

- Para obtener los registros de la tabla RH_EMPLEADO ordenados por apellido paterno, apellido materno y nombre

```
Select * from RH_EMPLEADO
order by STR_PATERNO, STR_MATERN0, STR_NOMBRE
```

- Para obtener los registros de la tabla RH_EMPLEADO agrupados por número de empleado y ordenados por el mismo campo, de esta manera podemos saber el número de veces que se repite la clave de un empleado y además con la instrucción having podemos saber solamente todas aquellas claves que se encuentran mas de una vez en la BD.

```
select INT_COD_EMPLEADO, COUNT(INT_COD_EMPLEADO)
from TYFACTURACION
group by INT_COD_EMPLEADO
having count(INT_COD_EMPLEADO) > 1
order by 2 DESC
```

1.1.5.3. Lenguajes de Programación

Un lenguaje de programación es un conjunto de instrucciones regidas por reglas que cumplen el fin de ser compiladas[13] para lograr hacer un programa que cubra las necesidades de un negocio, organización, institución, etc. Existen varios lenguajes de programación, COBOL, C++, JAVA, sin embargo nosotros nos abocaremos a JAVA[14] con su potente programación orientada a objetos y su máquina virtual. Los lenguajes de programación han venido evolucionando durante el tiempo, así como también las formas de programar, hoy en día podemos apreciar las innumerables herramientas que acompañan a los lenguaje de programación, tanto para el diseño, implementación y el diseño de interfaz, estas nos dan la oportunidad de mejorar nuestros programas haciéndolos más estéticos, y vaya que eso es importante en el mercado, sin embargo por mas importancia que tenga la apariencia de los programas no es mas que el simple cascarón, ya que atrás de todo ello existen rutinas especializadas y una metodología de programación bien definida, y de esto ultimo es de lo que hablaremos.

Cuando se habla del mundo de la computación se habla de Software que son programas, y hardware que son los componentes físicos (disco duro, memoria, etc.), en este apartado hablaremos del software, aquellos programas que hacen que el ordenador o computadora funcione, el ordenador trabaja con un lenguaje llamado lenguaje máquina, este se compone de bits, un bit[15] nos representa un 0 o un 1 (niveles de voltaje), por lo tanto todas las instrucciones que ejecuta el ordenador es por medio de 1 y 0, escribir programas solamente con estos dos elementos se vuelve muy complicado así que surgió un lenguaje llamado "Lenguaje ensamblador" que utiliza códigos mnemotécnicos[16] para indicarle al hardware las operaciones deseadas, estas instrucciones se ejecutan en un programa llamado "Ensamblador", el cual se encarga de interpretar las instrucciones para dar como resultado instrucciones en lenguaje máquina. La evolución tecnológica posteriormente nos muestra una manera más sencilla de escribir programas, y es cuando surgen los lenguajes de programación de alto nivel, esto significa que se alejan cada vez mas de la preocupación del tipo de procesador del ordenador, lo que no sucedía en los lenguajes ensambladores.

Los lenguajes de alto nivel son traducidos por un programa llamado "compilador" que es el que se encarga de traducir todas sus instrucciones a un lenguaje máquina. De esta manera la programación de los sistemas ha sido cada vez más rápida y más eficiente, ya que cada vez que pasa el tiempo tenemos mas y más herramientas de desarrollo, lo que significa que cada vez no vamos preocupando menos por los detalles de apariencia y de rescribir código, como sucede en la programación orientada a objetos, dónde podemos explotar la característica de la reutilización de código por medio de la herencia..

Los lenguajes donde se programaba de una manera estructurada llegaron a un límite cuando se enfrentaron a problemas mayores, los archivos de código fuente se volvían extremadamente extensos y era difícil llevar un control del mismo, así que ahora con la POO (utilizaré esta abreviación de aquí en adelante para referirme a la Programación Orientada a Objetos), nos encontramos con una reutilización de código y la organización del mismo en diferentes archivos que representan partes del mismo problema, y así estas, pueden aislarse y ser tratadas como una encapsulación de datos, este último concepto fue explicado el apartado "Clases y Objetos".

[13] Proceso de traducción de instrucciones de un lenguaje de alto nivel a instrucciones de lenguaje máquina.

[14] Lenguaje de Programación Orientado a Objeto

[15] Unidad mínima de almacenamiento en un ordenador

[16] mnemotécnico son palabras que se utilizan para recordar ciertas instrucciones mas complejas que ella misma.

JAVA

Cuando se decide desarrollar un sistema, dentro de las cosas que se eligen es el lenguaje de programación, ya que existen lenguajes especializados en áreas específicas, otros como JAVA nos ofrece un sin fin de funcionalidades.

Como ya mencioné con anterioridad, los lenguajes de alto nivel necesitan de un compilador que traduzca sus instrucciones a lenguaje máquina, pues JAVA es uno de ellos, solo que la gran diferencia que existe ante los demás lenguajes es que JAVA no solamente tiene un compilador, si no que también cuenta con un interprete llamado Máquina Virtual de JAVA, y esta al ser un programa que simula una máquina, puede ser instalada en cualquier ordenador, de tal manera que los programas escritos en JAVA puedan ser ejecutados en cualquier ordenador que contenga esta máquina virtual, y no tendremos que preocuparnos de la plataforma ni del tipo de procesador. En la Figura 1.32 podemos apreciar el proceso de compilación de un programa en JAVA.

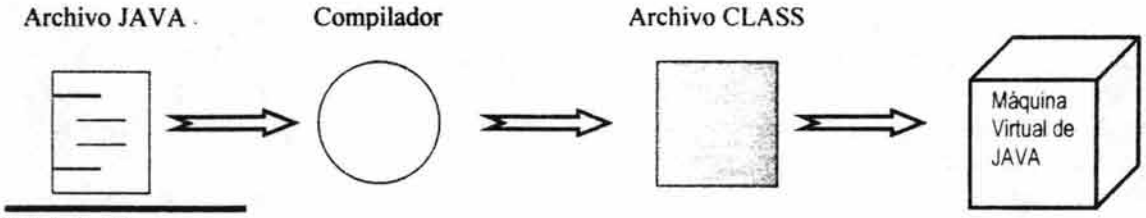


Figura 1.32
Proceso de compilación en JAVA

Este lenguaje fue desarrollado por Sun Microsystems en 1991, como parte de un proyecto de comunicación entre aparatos electrónicos, sin embargo cada aparato tenía un procesador diferente, entonces de aquí surgió la idea de crear un lenguaje de programación independiente del procesador, el lenguaje llamado en un principio como Oak..Posteriormente cuando surgió Internet los investigadores se dirigieron a crear un lenguaje en donde se pudieran crear aplicaciones para Internet, y así es como surgió JAVA.

JAVA es un lenguaje que está fundamentado en C++ y además de que es portable, es un lenguaje orientado a objetos.

1.1.5.4 Fundamentos de programación

Un programa está constituido por variables que pertenecen a ciertos tipos de datos, funciones, estructuras de control, clases, objetos, librerías creadas por el usuario, etc. Nosotros nos basaremos en el lenguaje JAVA para este apartado.

Para escribir un programa es necesario tener el conocimiento del lenguaje en el cual se va a escribir, y el planteamiento del problema, así como el diseño orientado a objetos en el caso de JAVA, también es necesario tener la lógica de programación, es decir, el conocimiento de implementación de las herramientas que nos ofrece el lenguaje.

Tipos de datos en JAVA

Tipo de dato	Longitud	Descripción
byte	8 bits	Para declarar datos enteros entre –128 y +127
short	16 bits	Para declarar datos enteros entre –32768 y +32767
int	32 bits	Para declarar datos enteros entre –2147483648 y +2147483647
long	64 bits	Para declarar datos enteros entre –9223372036854775808 y 9223372036854775807
char	16 bits	Para declarar datos enteros en Unicode(0 a 65535), del 0 al 127 corresponden a los caracteres ASCII
float	32 bits	Para declarar datos flotantes con una precisión aproximada de 7 dígitos
double	64 bits	Para declarar datos flotantes con una precisión aproximada de 16 dígitos
boolean		Con valores true o false constantes definidas como palabras clave en JAVA

Declaración de variables

Para declarar 3 variables del tipo de datos short

```
short x,y,z;
```

Para declarar 1 variable del tipo int inicializada por el valor de 0.

```
int contador = 0;
```

Para declarar dos objetos miembros de la clase String:

```
String Nombre = "Alberto";
String SedundoNombre = new String("Angel");
```

Operadores

Son los símbolos que utilizamos para manipular los datos.

Operadores Aritméticos	
Operador	Descripción
+	Suma
-	Resta
*	Producto
/	División
%	Módulo
Operadores relacionales	
<	Menor que
>	Mayor que
<=	Menor igual que
>=	Mayor igual que
!=	Diferente de
==	Igual que
Operadores lógicos	
&&	AND
&	AND. se evalúan los dos operándos
||	OR
|	OR. Se evalúan los dos operándos
!	NOT
^	XOR
Operadores unitarios (a un solo operando)	
~	Complemento a 1 (cambia ceros por unos y unos por ceros)
-	Cambio de signo
Operadores a nivel de bits (operándos enteros)	
&	AND
|	OR
^	XOR
<<	Desplazamiento a la izquierda, rellenando con ceros por la derecha
>>	Desplazamiento a la derecha, rellenando con el bit de signo por la izquierda
>>>	Desplazamiento a la derecha rellenando con ceros por la izquierda
Operadores de Asignación	
++	Incremento
- -	Decremento
=	Simple asignación
*=	Multiplicación más asignación
/=	División más asignación
%=	Módulo más asignación
+=	Suma más asignación
-=	Resta más asignación
<<=	Desplazamiento a la izquierda más asignación
>>=	Desplazamiento a la derecha más asignación
&=	AND sobre bit más asignación
|=	OR sobre bits más asignación
^=	XOR sobre bits más asignación

Sentencias de control

If

Permite tomar una decisión, y así ejecutar un bloque de sentencias, en el caso de que el resultado sea verdadero o falso, si es falso ejecuta el bloque de sentencias contenido en el else.

Ejemplo

```
If ( x < = 10 && y = 7 )
{
        bloque de sentencias 1;
}
else
{
        bloque de sentencias 2;
}
```

Dónde:

Si x es menor o igual al valor constante 10 y es igual al valor constante 7 entonces ejecuta el bloque de sentencias 1, de lo contrario ejecuta el bloque de sentencias 2.

switch

Permite tomar una decisión en base a múltiples opciones, con esta sentencia evitamos escribir varios if.

Ejemplo

```
switch( variable_x )
{
        case 1:
                bloque de sentencias 1;
                break;
        case 2:
                bloque de sentencias 2;
                break;
        default:
                bloque de sentencias 3;
                break;
}
```

Dónde:

La variable en cuestión es variable _x, si ésta es igual al valor constante 1, ejecuta el bloque de sentencias 1, sino, procede a comparar el siguiente caso, que es el valor constante 2, si es igual a este, ejecuta el bloque de sentencias 2, si no es igual a ninguno de los casos, ejecuta el bloque de sentencias 3, que se encuentra en el default. La palabra reservada break finaliza la ejecución de la sentencia.

while

Repite la ejecución de la sentencia contenida entre sus corchetes, mientras la expresión que se encuentra entre paréntesis sea verdadera.

Ejemplo

```
while( x < 100 )
{
        bloque de sentencias;
}
```

Dónde:
Mientras que el valor de la variable sea menor al valor constante 100, ejecuta el boque de sentencias.

do... while
Repite la ejecución de la sentencia contenida entre sus corchetes, mientras la expresión que se encuentra entre paréntesis sea verdadera.

Ejemplo

```
do
        bloque de sentencias;
while( x < 100 );
```

Dónde:
Mientras que el valor de la variable sea menor al valor constante 100, ejecuta el boque de sentencias. La diferencia entre el while y do while es que en el while se evalúa primero la expresión y después se ejecuta el bloque de sentencias, y en el do while, primero se ejecuta la sentencia y después se evalúa la expresión entre paréntesis.

for
Repite la ejecución de una sentencia el número de veces especificado.

Ejemplo

```
for ( x= 0; x <= 10; x + + )
{
        bloque de sentencias;;
}
```

Dónde:
La variable x comienza con valor igual a cero, incrementa en uno cada iteración[17], y así hasta que llega al valor de 11, cuándo sucede esto ya no ejecuta mas el bloque de sentencias.

break
La sentencia break finaliza la ejecución de las sentencias de control switch, while, do y for, es decir, pone fin al bucle[18].

[17] Se refiere a un ciclo de proceso. Dos iteraciones significa que el bloque de sentencias se ha ejecutado 2 veces
[18] Se le llama al proceso de repetición de una sentencia de control.

Ejemplo

```
for ( x= 0; x <= 10; x + + )
{
        bloque de sentencias;;
        x = x * 3;
        if ( x > 10 ) break; // salir del bucle.
}
```

Dónde:
Dentro del bucle si x es mayor al valor constante 10, sal del bucle

continue
Obliga a ejecutar la siguiente iteración.

Ejemplo

```
for ( x= 0; x <= 10; x + + )
{
        bloque de sentencias;;
        if ( ( x % 2 ) != 0 ) continue; // continuar a la siguiente iteración
        // Imprime el valor contenido en x
        System.out.println( x );
}
```

Dónde:

Dentro del bucle si el número es impar continúa la siguiente iteración, es decir solamente queremos obtener los números pares.

Métodos

Los métodos (también llamados funciones)no son mas que un conjunto de sentencias compuestas que se encuentran entre corchetes, y son ejecutadas cada vez que se manda a llamar al método, es decir, no es necesario escribir todo el código de las sentencias, si no que es suficiente con mandar a llamar por su nombre al método que las contiene.

Ejemplo

```
Static int Producto( int x, int y )
{
        return x * y;
}
```

Dónde
La función recibe como parámetros[19] dos valores que se almacenan en las variables x, y, estas se encuentran entre paréntesis, esta retornará un tipo de dato entero especificado al principio de su declaración, y esto se logra con la palabra reservada return[20]. Podemos apreciar el diagrama de esta función en la Figura 1.33

[19] Los valores de entrada a una función

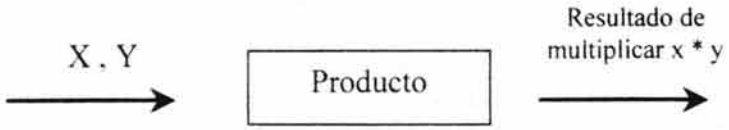


Figura 1.33
Método Producto

[20] El los valores de salida de una función o método.

1.1.5.5 Programación Orientada a Objetos en JAVA

En el apartado "Clases y Objetos", expuse una breve explicación sobre lo que son las clases y los objetos, ahora veremos la programación de los mismos en el lenguaje JAVA, haremos programas que nos darán las bases para programar en este lenguaje, nótese que antes de continuar debemos de percatarnos que en realidad hayamos entendido el diseño y la lógica Orientado a Objetos, ya que es de primordial importancia porque apartir de aquí utilizaremos este conocimiento para aplicarlos e implantar nuestros diseños en el lenguaje de programación JAVA.

Primer Programa en JAVA: HOLA.java

Veamos el siguiente código fuente[21]:.

```
public class HOLA
{
        public static void main (String[] args)
        {
                //Imprimimos en pantalla
                System.out.println("Mi primer programa en Java");
                System.out.println("Desde la función principal");
        }
}
```

El programa expuesto no hace mas que imprimir dos líneas de texto. Dónde:

Las palabras que se encuentran en negritas se llaman palabras reservadas, estas son palabras del propio lenguaje. Ahora veámoslo por partes:

public class HOLA
Se declara una clase llamada HOLA, esta clase es de alcance público.

public static void main (String[] args)
Se declara un método o función llamado main, de alcance público, del tipo **static**, no retornará ningún valor (**void**) y puede recibir parámetros del tipo **String,** esta función es la función principal, es decir, el compilador lo primero que hará es buscar dicha función, y es aquí dónde comienza a ejecutar el código fuente.

System.out.println("Mi primer programa en Java");
Se imprime la cadena de caracteres especificadas por medio de la función println, este es el cuerpo de la función main encerrado entre { }.

Y finalmente con ' // ' indicamos comentarios al código, es texto que nos ayuda a explicar el comportamiento del código, este no es tomado en cuenta por el compilador.

Ahora agreguémosle un método a nuestro programa:

[21] Es el conjunto de instrucciones escritas en un lenguaje.

```
private void Mensaje()
{
        System.out.println("Ahora mando un Mensaje");
        System.out.println("desde un método Miembro de la Clase HOLA");
}
```

Es un método de alcance privado que no retorna ningún valor y tampoco recibe parámetros, y su función es imprimir dos líneas de texto. Para que esto sea posible tenemos que agregar algunas líneas de código a la función principal.

```
public static void main (String[] args)
{
        HOLA MiMensaje = new HOLA();

        //Imprimimos mensajes
        System.out.println("Mi primer programa en Java");
        System.out.println("Desde la función principal");
        MiMensaje.Mensaje();
}
```

HOLA MiMensaje = **new** HOLA();
Se crea un objeto llamado MiMensaje miembro de la clase HOLA.

MiMensaje.Mensaje();
Por medio del Objeto MiMensaje creado, se manda a llamar al método Mensaje.

Herencia

Ahora hagamos un programa más interesante, utilizando herencia, para esto plantearemos un problema:

> Se desea informar a todos los empleados la cantidad en pesos a cobrar por concepto de aguinaldo, suponiendo que todos los empleados tienen el mismo sueldo, pero los días de aguinaldo a cobrar son: 23 días para el personal de confianza, y 60 días para el personal sindicalizado. El salario diario de cada empleado es de 567.3 pesos.

Para el problema planteado haremos el siguiente diseño de jerarquía de clases Rational Rose. Ver Figura 1.34

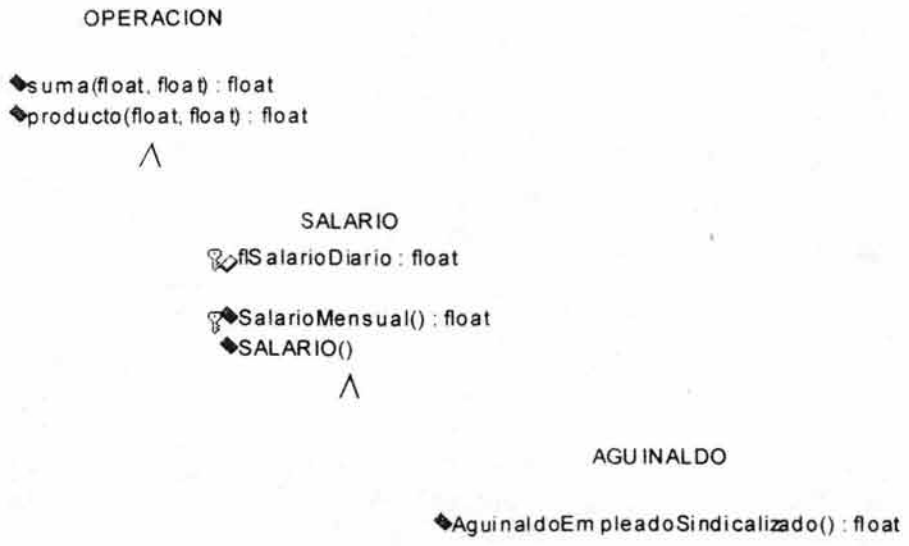


Figura 1.34
Jerarquía de clases del ejemplo de herencia

Como podemos observar la clase AGUINALDO hereda los atributos y métodos correspondientes de la clase SALARIO, y esta a su vez los hereda de la clase OPERACIÓN, ésta última será nuestra superclase.
La cual se encargará de contener los métodos se suma y producto, los cuales recibirán como parámetros dos valores flotantes, retornando así el resultado correspondiente por medio de la instrucción **return**, veamos el código de esta clase:

```
public class OPERACION
{
        public float Suma (float flOperandoA, float flOperandoB)
        {
                return flOperandoA + flOperandoB;
        }
        public float Producto (float flOperandoA, float flOperandoB)
        {
                return flOperandoA * flOperandoB;
        }
}
```

La subclase inmediata de OPERACIÓN es SALARIO la cual se encarga de almacenar el salario diario de todos los trabajadores, inicializando su atributo con ese valor, también contiene un método llamado SalarioMensual, el cual se encarga de multiplicar el salario diario por 30 días:

```
public class SALARIO extends OPERACION
```

```
{
        protected float flSalarioDiario;

        public SALARIO()
        {
                flSalarioDiario = 567.3F;
        }

        protected float SalarioMensual()
        {
                return flSalarioDiario * 30;
        }
}
```

Por último tenemos a la clase AGUINALDO, la cual resuelve en parte el problema, ya que su método llamado AguinaldoEmpleadoSindicalizado() se encarga de retornar el valor en pesos del aguinaldo del personal sindicalizado. En esta clase podemos apreciar la herencia, ya que esta contiene sus propios miembros que es solamente el método, y además contiene los métodos suma y producto los cuales heredó de la clase OPERACIÓN, el método SalarioMensual y atributo flSalarioDiario que heredó de la clase SALARIO, así que cuando se crea el objeto llamado CalculaAguinaldo miembro de la misma clase, por medio de él podemos acceder a los elementos mencionados, y así mostrarlos en pantalla.

```
public class AGUINALDO extends SALARIO
{
        public float AguinaldoEmpleadoSindicalizado()
        {
                float flAguinaldoTotal;
                float flSalario;
                AGUINALDO CalculaAguinaldo = new AGUINALDO();
                flSalario = CalculaAguinaldo.flSalarioDiario;
                return CalculaAguinaldo.Producto(flSalario,60);

        }
}
```

En la clase principal llamada PRESTACIÓN encontramos que creamos otra instancia de la clase AGUINALDO, haciendo lo mismo que en el método AguinaldoEmpleadoSindicalizado de esa misma clase, solo que ahora obtenemos el aguinaldo del personal de confianza enviando como parámetro al método producto el salario diario y los 23 días correspondientes. La función println o print toman el valor retornado por el método AguinaldoEmpleadoSindicalizado().

```
import java.sql.*;
import java.awt.* ;
public class PRESTACION
{

        public static void main(String[] args)
        {
                float flAguinaldoTotal;
                float flSalario;
                AGUINALDO CalculaAguinaldo = new AGUINALDO();
                flSalario = CalculaAguinaldo.flSalarioDiario;
                flAguinaldoTotal = CalculaAguinaldo.Producto(flSalario,23);

                System.out.print("El Aguinaldo para el personal de Confianza:");
                System.out.println(flAguinaldoTotal);
                System.out.print("El Aguinaldo para el personal Sindicalizado:");
                System.out.print(CalculaAguinaldo.AguinaldoEmpleadoSindicalizado());

        }
}
```

1.2. El Departamento de Recursos Humanos

En el tiempo en que he estado en diferentes empresas, muy cerca del Departamento de Recursos Humanos, me he dado cuenta que este es extremadamente importante para una empresa, conforme a su desarrollo, RH (utilizaré estas siglas para referirme al Departamento de Recursos Humanos) debe de tener una mejor organización ya que entre mas empleados existe una mayor cantidad de información, de la cual es muy fácil perder el control. RH es la organización, planeación de la información de todos y cada una de las personas que laboran en la empresa, y tener una buena gestión de esta información contribuye al buen desarrollo de la misma.

RH contempla muchas cosas, desde declaraciones ante el gobierno, como el control interno de niveles puestos, realizando estudios constantes sobre la actitud de las personas, estimando una atmósfera de trabajo, mediante estadísticas, cuestionarios, observación, producción, utilidades, es decir, como se encuentre la gente, así se encontrará la empresa. La misión de RH a mi parecer es preocuparse por los empleados, y de esta forma automáticamente se trabaja por la empresa, es un efecto totalmente reflejado en el bienestar de la empresa. RH se realiza preguntas como:

- ¿Contamos con la cantidad de personas que necesitamos?
- ¿Contamos con personal capacitado?
- ¿A la gente le gusta lo que hace en esta empresa?
- Todas las personas que laboran en nuestra empresa ¿Cumplen con el perfil del puesto?
- ¿Nos encontramos actualizados en los exámenes que aplicamos a los aspirantes?
- ¿Tenemos suficientes contactos con otras empresas para intercambio de personal?
- ¿El tiempo que se invierte para un ingreso es el pertinente?

Y muchas preguntas mas, de esta manera RH tiene el conocimiento de " como va la empresa ", los resultados son irrefutables, las utilidades, el objetivo de la empresa, resultado global del esfuerzo conjunto.

Muchas empresas no le dan la importancia que tiene RH, y no se han dado cuenta o no lo han querido reconocer, pero ha sido una de las causas de su fracaso rotundo, por no saber administrar a la gente, preocuparse por ella.

RH viene siendo para la empresa los cimientos, si RH está mal, la empresa está mal, planteémoslo así, ¿Quién es la empresa?, ¿Un edificio?, ¿Una firma?, ¿Un logo?, no, no es nada de eso, la empresa es la gente que la forma, y por la gente la empresa existe. Cuando sucedió la Revolución Industrial las máquinas sustituyeron a la gran parte de la gente que trabajaba, sin embargo la empresa no dejó de necesitar a la gente que operaba a esas máquinas, pienso que por más que avancemos en la tecnología la gente siempre será la más importante, son lo que hacen posible todo, los empresarios que no han entendido esto son los que han fracasado definitivamente, o que solo quieren mano de obra barata, es cierto, al principio obtienen grandes ganancias, sin embargo, a la larga, la gente reclama, empieza hacer sindicatos y huelgas con todo el derecho, y la empresa empieza a declinarse hasta que toca suelo.

Ahora bien, la decisión de desarrollar o comprar un sistema de información para que agilice la administración del personal es una decisión que tienen que tomar todos los empresarios de micro, medianas y grandes empresas. El sistema que se expone en el presente documento cubre todas las necesidades de administración de personal tanto para una micro empresa de 30 empleados, hasta una macro empresa que tenga mas de 20,000 empleados laborando, gestionando toda la información, así como el cálculo de la nómina,

llevando históricos de toda su información, de tal manera que RH podrá tener acceso a un historial detallado del empleado.

En Sección Amarilla existe cada año reuniones con el Sindicato, dónde platican los aumentos que existirán para todos los empleados, estos aumentos se dan en porcentaje, aplicado directo al sueldo diario como también a las prestaciones. Hablando un poco de estas últimas, existen durante el año diferentes promociones de ventas de productos a los empleados, por medio de los acuerdos que tiene la empresa con otras empresas que ofrecen sus productos, siendo así, Sección amarilla paga todos los productos escogidos por los empleados, y a estos se les va descontando en cierto número de quincenas, por lo tanto los descuentos que puede tener un empleado son muy variables, estos van afectando a las demás prestaciones como Fondo de ahorro, RH es el encargado de controlar todas estas situaciones, llevando un histórico de información.

Capítulo II
Análisis de los Requerimientos

2.1. Estudio Preliminar y Análisis Costo - Beneficio.

2.2 Análisis Costo - Beneficio

Base de Datos

La implementación de un Sistema de Información en SQL Server no requiere tanta dedicación como sistemas montados en otras Bases de Datos como ORACLE o Sysbase, en el caso de la empresa Sección Amarilla, cuenta con un equipo administrador de base de datos dedicado a administrar todas las bases de datos existentes en la empresa, por lo tanto podemos decir que tenemos cubierto la administración de nuestra base de datos.

Entonces, los costos de capacitación para manejar SQL Server es CERO.

En cuanto al licenciamiento de SQL Server tampoco tenemos problemas ya que se cuentan con las licencias de SQL Server para más de 50 usuarios, y en realidad el software de servidor de SQL Server ya se encuentra instalado en los servidores, platicando con el departamento de infraestructura solo es cuestión de crear la instancia de Base de Datos correspondiente a Recursos Humanos, y el software del cliente pues solo será para un solo desarrollador.

Entonces, los costos de Licenciamiento para SQL Server es CERO.

Sistema Operativo

Para el sistema operativo al igual que el software de base de datos ya se encuentra instalado en el servidor (Windows NT Server) y en los clientes (Windows 2000). Y además se cuentan con las licencias suficientes para soportar el sistema. El cual será utilizado por 20 usuarios aproximadamente.

Entonces, los costos de Sistema Operativo es CERO.

Servidor de Aplicaciones

Como estamos hablando que el sistema será utilizado por 20 usuarios, es suficiente utilizar un Servidor Apache en una de sus versiones de software libre, ya que no requerimos un arreglo complejo de servidores Web, siendo que el sistema se encontrará en una Intranet, y este servidor Apache será dedicado únicamente a este sistema.

Entonces, los costos de Licenciamiento por servidor Web es CERO.

Desarrolladores

En realidad este sistema será desarrollado por una sola persona, por lo tanto los honorarios del Analista serán:

Actualmente se cuenta con un sistema que no cumple con el total de requerimientos del departamento, incluso, tienen una estimación de mantenimiento sin garantía durante los próximos 10 meses para desarrollar todos aquellos módulos que no sean desarrollado y optimizar el proceso de nómina actual.

A continuación presentaré un comparativo de costos, sobre lo que costaría dar mantenimiento al sistema actual y desarrollar el nuevo sistema.

Sistema Actual		
Mantenimiento sin Garantía	Costo Hora Hombre	250.00
	8 al Día	2,000.00
	20 Días al Mes	40,000.00
	10 Meses	400,000.00
Total por mantenimiento a Nuevos Requerimientos		400,000.00

Para el sistema nuevo se estima un mantenimiento con garantía durante 12 meses, esto quiere decir que cualquier error o deficiencia en el sistema, se corregirán sin costo alguno, si surge un nuevo requerimiento durante ese tiempo, estos nuevos desarrollos tendrían un costo, esto es: Mantenimiento sin garantía. Como no es posible saber qué requerimientos habrá en el futuro, estos costos no se calcularán solo se expondrá.

Sistema Nuevo		
Análisis y Desarrollo	Costo Hora Hombre	125.00
	8 al Día	1,000.00
	20 Días al Mes	20,000.00
	12 Meses	240,000.00
Implantación	Costo Hora Hombre	130.00
	8 al Día	1,040.00
	20 Días	20,800.00
Mantenimiento - Garantía	Costo Hora Hombre	0.00
	10 al Mes	0.00
	12 Meses	0.00
Mantenimiento sin Garantía	Costo Hora Hombre	170.00
	Total de Costo	260,800.00

Ahora veamos la tabla comparativa:

Comparativo	
Costo Nuevo Sistema	260,800.00
Costo Mantenimiento Sistema Actual	400,000.00
Diferencia	**139,200.00**

Por lo tanto, el desarrollar un nuevo sistema en el tiempo especificado es más económico que dar mantenimiento al sistema actual

Hardware

Hoy día, Sección Amarilla cuenta con la infraestructura adecuada para el Desarrollo de un Sistema de Información para el Departamento de Recursos Humanos, el servidor que se utilizará cuenta con dos microprocesadores a 996 Mhz cada uno, 2 GB en memoria RAM y 600GB en disco duro. Así como también 15 pc´s clientes con microprocesador pentium IV con 256 MB en memoria RAM y 10 GB en disco duro. Todo el equipo se encuentra conectado a una red de area local.

Software

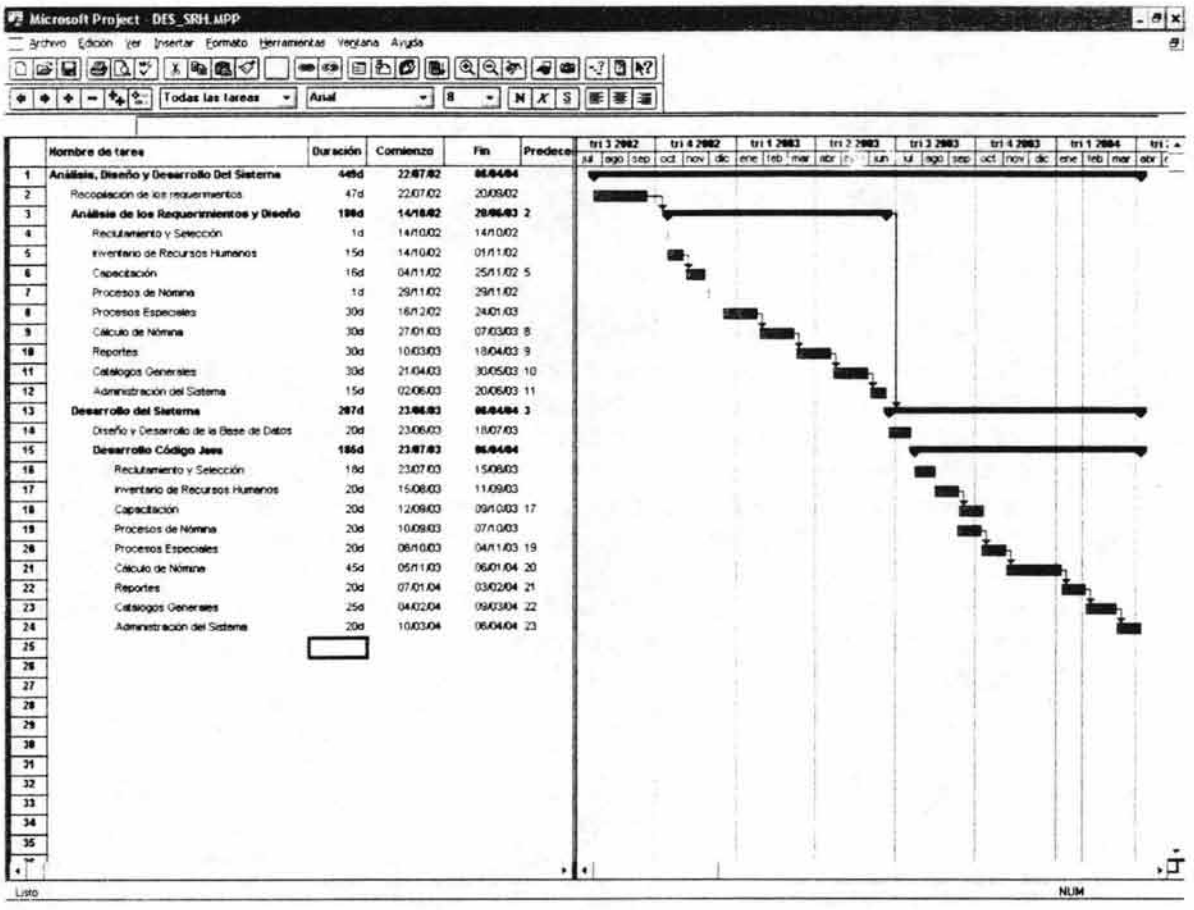
En cuanto al software, el servidor cuenta con Windows 2000 Advanced Server, SQL Server 2000 de Microsoft y servidor Apache 1.3.22, los clientes cuentan con Windows 2000. Para el desarrollo se cuanta con las herramientas jdk1.3 de Sun MicroSystems, jDeveloper de Oracle, Rational Rose, Power Designer.

Recursos Humanos

Bien, este proyecto ha sido requerido por el Departamento de Recursos Humanos, por otro lado el Departamento de Sistemas asignará a una sola persona para desarrollar el Sistema, ya que el requerimiento no es urgente se acordó con recursos Humanos que se entregará en un tiempo estimado de año y medio.

Planeación de Actividades

La planeación de actividades se realizó en Microsoft Project 4.0, la podemos ver en la siguiente figura, o bien en el archivo RH.MPP.


Microsoft Project - DES_SRH.MPP
Archivo Edición Ver Insertar Formato Herramientas Ventana Ayuda
Todas las tareas
Arial
Nombre de tarea
Duración
Comienzo
Fin
Predece
1 Análisis, Diseño y Desarrollo Del Sistema 448d 22/07/02 06/04/04
2 Recopilación de los requerimientos 47d 22/07/02 20/09/02
3 Análisis de los Requerimientos y Diseño 186d 14/10/02 20/06/03 2
4 Reclutamiento y Selección 1d 14/10/02 14/10/02
5 Inventario de Recursos Humanos 15d 14/10/02 01/11/02
6 Capacitación 16d 04/11/02 25/11/02 5
7 Procesos de Nomina 1d 29/11/02 29/11/02
8 Procesos Especiales 30d 16/12/02 24/01/03
9 Cálculo de Nómina 30d 27/01/03 07/03/03 8
10 Reportes 30d 10/03/03 18/04/03 9
11 Catalogos Generales 30d 21/04/03 30/05/03 10
12 Administración del Sistema 15d 02/06/03 20/06/03 11
13 Desarrollo del Sistema 267d 23/06/03 06/04/04 3
14 Diseño y Desarrollo de la Base de Datos 20d 23/06/03 18/07/03
15 Desarrollo Código Java 186d 23/07/03 06/04/04
16 Reclutamiento y Selección 18d 23/07/03 15/08/03
17 Inventario de Recursos Humanos 20d 15/08/03 11/09/03
18 Capacitación 20d 12/09/03 09/10/03 17
19 Procesos de Nómina 20d 10/09/03 07/10/03
20 Procesos Especiales 20d 08/10/03 04/11/03 19
21 Cálculo de Nómina 45d 05/11/03 06/01/04 20
22 Reportes 20d 07/01/04 03/02/04 21
23 Catalogos Generales 25d 04/02/04 09/03/04 22
24 Administración del Sistema 20d 10/03/04 06/04/04 23
tri 3 2002
tri 4 2002
tri 1 2003
tri 2 2003
tri 3 2003
tri 4 2003
tri 1 2004
Listo
NUM

2.2. Planteamiento del Problema.

Actualmente Sección Amarilla (Anuncios en Directorios) utiliza un sistema llamado TESS, sobre Novell, este sistema se ha venido utilizando desde hace 10 años con esporádico soporte, actualmente es difícil encontrar en la empresa GMP(creadores de TESS) soporte para TESS, ya que no se encuentra la gente que lo hizo, sin mencionar el lenguaje en el que se realizó. En dicho sistema se almacena Información sobre los empleados y casi todos los procesos de nómina existentes, ya que se encuentra delimitado, en algunos procesos ya es nulo, esto provoca que el usuario realice estos procesos utilizando diversas herramientas como Microsoft Excel, esto conlleva a mayor tiempo de inversión así como la desorganización de las hojas de cálculo. Por lo tanto este sistema solo abarca el módulo de nómina en un 70%.

El proceso de Reclutamiento se realiza por medio de formatos impresos, estos se archivan físicamente, es decir no se cuenta con un sistema dónde puedan capturar toda esta información, la captura de la información del aspirante se realiza una vez que ya se encuentra aceptado como miembro de la empresa(en el sistema TESS). La estructura organizacional se encuentra definida en documentos PPT, los cuales no se encuentran actualizados, por la gran variedad de movimientos que existen. En cuanto a capacitación, el control de todos los cursos impartidos y por impartir se encuentran en hojas de Excel, sin embargo no se lleva una correcta gestión de los mismos, ya que es muy difícil obtener históricos de cursos durante el año, o años anteriores.

Como podemos ver, la empresa en cuanto al departamento de Recursos Humanos, nunca ha contado con un sistema que integre todos sus módulos sin depender de las hojas de cálculo de Excel, es por eso que existe demasiada fuga de información.

2.3. Objetivo General

Sistematizar [22] las actividades del Departamento de Recursos Humanos de la Empresa Anuncios en Directorios S.A. de C.V (Sección Amarilla), para elevar su nivel de administración de personal.

2.4. Objetivos específicos

- ☑ **Sistemas Comerciales**. Investigar y Estudiar Software ya creado, preguntando a los usuarios su funcionalidad y alcance para obtener experiencia.
- ☑ **Reclutamiento**. Analizar [23] el sistema metodológico de reclutamiento, para diseñar el flujo de información.
- ☑ **Inventarios**. Analizar diferentes métodos de administración para el control de Inventarios.
- ☑ **Procesos de Nómina.** Analizar el cálculo de percepciones y deducciones, para diseñar el proceso de automatización y ejecución de formulas en el proceso de nómina.
- ☑ Identificar cada uno de los procesos de Seguridad Social y Obligaciones Fiscales en la República Mexicana para optimizar el cumplimiento de estos rubros.

[22] Organizar un conjunto de elementos, coordinando la relación que existe entre ellos, para lograr un objetivo.
[23] Realizar un estudio detallado de alguna situación u objeto.

- ☑ Reconocer los procedimientos correspondientes a la realización de préstamos, y los tipos de préstamos con los tipos de amortización existentes, para diseñar el flujo de información.

- ☑ Analizar los procesos especiales; participación de utilidades (PTU), aumentos saláriales, cancelaciones para realizar la correspondiente interfase con el proceso de la nómina.

- ☑ Recopilar todos los formatos de los reportes y listados requeridos por las instancias gubernamentales para trazar los reportadores fijos y variables.

- ☑ **Administración del Sistema** . Analizar el flujo de información a nivel módulo para diseñar la Administración del Sistema.

- ☑ Identificar aquellos módulos que no deben de estar disponibles a todos los usuarios para diseñar la seguridad del sistema.

- ☑ Analizar los datos existentes en las empresa, para diseñar la Base de Datos.

2.5. Delimitación, Justificación del Tema

La herramienta comercial llamada Excel de Microsoft [24], ha sido utilizada para cubrir gran parte de los procesos de nómina de esta empresa, esta herramienta es efectiva hasta cierto punto, ya que cuando hablamos de una cantidad mayor de empleados, esta herramienta será obsoleta sin no se recurre a varios trucos para poder utilizarla, ya que esta herramienta no soporta mas de 65,536 registros, ahora si hacemos un análisis de las nóminas e cualquier empresa que contenga mas de 1,000 empleados, es evidente que no es posible continuar en Excel, ya que viendo específicamente la nómina de Agentes de Ventas de la Sección Amarilla, estamos hablando que cada quincena se generan 16 movimientos por empleado, la nómina de Agentes de Ventas cuenta actualmente con aproximadamente 897 empleados, por lo tanto:

Movimientos = Num de registros = 897 * 16 = 14,352

Ahora analizando la nómina de Administrativos, se generan cada quincena, promedio de 14 movimientos por empleado, siendo que se cuentan con 1,100 empleados, entonces

Movimientos = Num de registros = 1100 * 14 = 15,400

Y para la nómina de reparto tenemos un total de 1740 empleados aproximadamente, ya que aquí la mayoría son eventuales, tienen 5 movimientos promedio entonces:

Movimientos = Num de registros = 1740 * 5 = 8,700

Si sumamos los movimientos de las tres nóminas:

Total de movimientos de las 3 nóminas = 38,452

Esto es en una quincena, ahora si tomamos en cuenta un año de nómina:

Total de movimientos en un año = 922,848

Estos cálculos son sin tomar en cuenta el crecimiento de la población de empleados en la empresa, de lo contrario esto se elevaría a mas de 1,000,000 de movimientos anuales, estos se incluyen en todo un cálculo que es el impuesto anual, esto quiere decir que no es posible el manejo de esta cantidad de registro en Excel, ahora tampoco estamos tomando en cuenta todos los de mas movimientos que se deben de generar como los históricos de sueldos, los finiquitos, etc, estamos hablando de una gran cantidad de registros.

Por otra parte el sistema de nómina que está utilizando Agentes de Ventas, se ha detectado que entre mas información exista en la base de datos este se hace mas lento en el cálculo de la nómina y la generación de los salarios integrados [25], entonces, si hablamos de un crecimientos de la empresa, no le doy mas de 1 año de vida al sistema. Este Sistema fue producto de un gran tiempo de desarrollo[26] y una completa desorganización en el diseño, sin mencionar que no hubo análisis alguno y la rotación de programadores [27] fue excesiva durante mas de 2 años.

---

[24] Software creado por la empresa Microsoft, y es una hoja de cálculo. Paquete Comercial.

[25] Salarios producto de un cálculo en base al número de prestaciones

[26] Acción de realizar físicamente el proyecto expuesto, después de un análisis

[27] Programador o Desarrollador. Persona que tiene el conocimiento del análisis y desarrollo de un sistema computacional en un lenguaje de programación.

Así que la solución de desarrollar un nuevo sistema es totalmente viable, ya que tomando en cuenta los puntos que se acaban de mencionar, la cantidad de registros no tendría problema alguno, porque absolutamente toda la información se encontrará en una Base de Datos[28] la cual puede ser, dependiendo de la infraestructura de la empresa en cuanto a SGBD[29] : ORACLE, SQL Server o Informix, ya que estos manejadores son muy potentes, específicamente ORACLE tiene la capacidad de almacenar Billones de registros así que no tendrá mayor problema para trabajar. El cálculo se realizará con código encapsulado en un objeto y se recurrirá a una rutina especial para este, de tal manera que el tiempo de cálculo y la administración de información será mínimo.

No se tendrá que recurrir a Excel para la generación de reportes, ya que toda esta información se encontrará en la base de datos y se contará con un reporteador especial en el cual se podrá obtener información sin restricción alguna, claro las restricciones serán los permisos que se estipulen para saber qué persona tendrá acceso a esa información, pero eso será parte del módulo de seguridad incluido en el sistema, así que los usuarios podrán consultar sin mayor problema históricos de meses, incluso de años atrás, con la fiabilidad de que la información obtenida es correcta, aun mejor podrán obtener reportes completos en el formato que ellos deseen y con las sumatorias elegidas, así tendremos velocidad de cálculo, fiabilidad, y comodidad de reportes, y el crecimiento de la empresa no será mayor problema, la única limitante de almacenamiento de información es la capacidad del servidor que esta fuera del alcance del sistema.

Se podrán calcular finiquitos y liquidaciones, impuesto anual y conceptos especiales que vayan surgiendo en el transcurso del tiempo, ya que el usuario tendrá la facilidad de crear formulas las cuales tendrán una gran variedad de operándos[30] , me refiero a que será posible realizar cálculos con elementos de la base de datos como todos los tipos de salario que se manejan y también se podrán hacer cálculos con las fechas correspondientes a los empleados como su fecha de ingreso, de planta, de baja, de reingreso, etc, estos nos abre un gran campo de posibilidades ya que en los cálculo no se estará limitando en nada, y así cuando surga un nuevo concepto que utilice los valores mencionados, no existirá mayor problema que solo crear las formulas correspondientes de una manera lógica, y listo tenemos nuestro cálculo de n conceptos. Por si fuera poco dentro de las formulas tendrán la opción de que los operándos sean condiciones, esto rompe con el esquema anterior, ya que anteriormente si surgía un concepto nuevo, el cual venía constituido por condiciones, tenían que realizar de nuevo una programación y modificar todo el ejecutable, ahora el mismo usuario podrá crear las formulas con estas condiciones sin limitación alguna.

Se contará con una bitácora de cálculo la cual informará el trayecto de cálculo del concepto seleccionado, así el usuario podrá consultar paso a paso el cálculo que se realizó y hacer las correcciones pertinentes, estas correcciones podrán realizarse una vez terminado el cálculo, y estas modificaciones serán responsabilidad de la persona que las modifica, y por cada actualización que realice el usuario, se guardará un histórico de modificaciones, donde se podrá consultar quién modificó qué y cuándo lo hizo.

El usuario tendrá la facilidad de asignar el préstamo en la fecha correspondiente, y llevar los saldos con sus intereses, así mismos se tendrán los informes requeridos específicamente de este movimiento con el formato requerido. Todo el procedimiento de la presentación de los salarios al IMSS, desde el cálculo de los salarios variables hasta la generación del archivo que insertan utilizando el programa del SUA. Podrán consultar todos los históricos de sueldos con prestaciones y sin prestaciones, incluyendo el histórico de puestos, es decir, el usuario podrá consultar toda la trayectoria del empleado trabajando en la empresa y la información real de los empleos

[28] Conjunto de datos organizados en tablas relacionales

[29] SGBD. Sistema Gestos de Bases de Datos

[30] Elemento perteneciente a una fórmula z = operándoA + operándoB

anteriores. También se podrán almacenar todos aquellos candidatos actuales, hasta que el usuario especifique el tiempo de almacenamiento del mismo.

Existirá un módulo dónde el usuario capturará los datos del candidato, incluyendo los resultados de los exámenes correspondientes, y el sistema retornará un resultado realizando una comparativa de perfiles, y haciendo un estudio de las respuestas que dio este, y en base a todo esto, este resultado será un informe sobre si el candidato es ideal al puesto solicitado. Lo cual no existe actualmente.

Aparte de que el sistema cubrirá absolutamente todos los requerimientos que existen en la nómina y contablemente lo que le concierne a esta, cubrirá también todo lo que resta Recursos Humanos, el cuál no ha contado con un sistema que gestione[31] todas sus actividades como tales, el Sistema Propuesto cubrirá desde los organigramas, dónde se podrán imprimir organigramas completos de la base de datos, la cual podrá ser actualizada por los usuarios de nómina y todos lo usuarios permitidos, así cuando hagan un cambio de puesto, esta actualización influenciará en los organigramas, siendo que actualmente estos organigramas los tienen fijos, y solo son de algunos niveles, no abarcan a toda la empresa, y son muy difíciles de actualizar, ya que son requeridos en un tiempo mínimo.

El sistema necesita información generada por el sistema ADAMS [32] y de todas las agencias en el proceso del cálculo de la nómina.

[31] Administrar, coordinar a todos los elementos incluidos en un entorno.
[32] Sistema de gestión de contratos y anuncios de la Sección Amarilla.

2.6. Recopilación de los Requerimientos.

Para la obtención de toda la información necesaria para nuestro proyecto, se utilizarán varias estrategias:

Tomando en cuenta que tenemos el total acceso al sistema actual, tanto externa como internamente,, se realizará un estudio sobre el mismo, enterándonos de los procesos actuales, siendo estos fácilmente visibles, esto se logrará abriendo archivo por archivo generado por el sistema, recorriendo el código fuente [33] de aquellos principales módulos, sin interesarnos todos los catálogos, se realizará un seguimiento de ejecución paso por paso, a nivel función u objeto [34] dependiendo como se encuentre hecho el sistema. Con esto podemos armar los diagramas de flujo[35] correspondientes y de esta manera interpretar el flujo correcto de información, claro que este procedimiento se realizará sobre los módulos que esté funcionando correctamente y estén satisfaciendo las necesidades del usuario. Con esto nos ahorraremos algunas entrevistas e investigación sobre la información que necesitamos.

Por otra parte, la recopilación de los formatos, formas fiscales y todo lo que corresponda a la obligaciones de la empresa frente a las disposiciones oficiales, se realizará mediante visitas a las instancias gubernamentales, cubriendo todos los requerimientos que realiza el gobierno a la micro, mediana y gran empresa.

La información recopilada en el momento se organizará de la siguiente manera: Conforme se reciba la información se identificará automáticamente guardándola en carpetas previamente rotuladas con los nombres de los módulos existentes en el departamento de Recursos Humanos, ya sean folletos, reglamentos, manuales, formatos, diagramas de flujo, jerárquicos, unidades magnéticas de almacenamiento, y toda aquella información explicada de palabra y plasmada en hojas por medio de diagramas de estado y notas importantes, así como en unidades de cintas de audio. Para la transferencia de información robusta, se utilizarán pc's portátiles dónde podrán conectarse a cualquier nodo de red [36] de la empresa por medio del cable UTP nivel 3[37].

[33] Conjunto de instrucciones que interpretadas se convierten en un programa.
[34] Entidad abstracta en el entorno de programación de ordenadores
[35] Diagrama regido por reglas especificas, se usa para representar la lógica de un proceso.
[36] Entrada dónde es posible realizar una conexión e incorporar un ordenador a toda una red
[37] Tipo de cable de red

2.7. Análisis de los **Requerimientos del Departamento de Recursos Humanos**

2.7.1 Análisis del Módulo de Reclutamiento y Selección

El objetivo de este módulo es "Proveer a la empresa del factor humano con el nivel y calidad que demandan sus objetivos en las gerencias", así es como se encuentra en los documentos de A.D.S.A.

Proceso de Reclutamiento

a. Solicitud para cubrir una vacante por parte de las divisiones a la Gerencia de Integración y Desarrollo.
b. Convocar al personal interno para cubrir esa vacante.

Las causas por las que se puede dar una vacante son:

a. Renuncia
b. Baja
c. Cambio
d. Plaza nueva
e. Temporalidad

Tipos de vacantes:

a. Sindicalizados Tiempo determinado.- Se presenta por sustitución de una persona en su ausencia temporal
b. Sindicalizados Obra terminada.- Esto es cuando se van a realizar trabajos extraordinarios.
c. Confianza Tiempo determinado.- Se presenta por sustitución de una persona en su ausencia temporal
d. Confianza Obra terminada.- Esto es cuando se van a realizar trabajos extraordinarios.

Para la selección son indispensables los siguientes elementos:

- Solicitud de empleo
- Evaluación psicométrica
- Perfil General de habilidades del puesto

Una vez realizadas las evaluaciones mencionadas, el auxiliar de personal realiza el Reporte de Evaluación, el cual es la integración de los resultados obtenidos a través de los exámenes técnicos, psicométricos y la entrevista con el candidato vaciados en el gráfico del perfil general de habilidades del puesto. La contratación se lleva acabo una vez que se han cumplido con los requisitos antes mencionados, y se requiere la documentación que a continuación se menciona:

- Constancia del IMSS
- Constancia del último grado de estudios
- RFC
- CURP
- AFORE
- Cartilla Liberada
- Comprobante de domicilio
- Cartas de recomendación

- Acta de nacimiento
- INFONAVIT
- Identificación oficial
- Constancia de ingresos y retención de impuestos del año vigente (Forma 81)
- Fotografías
- Examen médico

Los movimientos en los puestos, están regidos por los cambios y promociones. Un cambio de puesto puede ser que no implique un aumento en el salario del empleado, si no el desarrollo de funciones distintas a las ve venía realizando con anterioridad, o un cambio de plaza. Una promoción es un cambio de puesto, que implica aumento en responsabilidades e ingreso del empleado.

Una vez que el empleado ingresó a la empresa se realizará una evaluación mensual, esto es para monitorear el desempeño del empleado durante ese tiempo, esto servirá para tomar la decisión de otorgar la planta. A continuación se presenta el contrato que es firmado por los representantes de la empresa y por el empleado:

CONTRATO INDIVIDUAL DE TRABAJO POR OBRA DETERMINADA QUE CELEBRAN POR UNA PARTE ANUNCIOS EN DIRECTORIOS, S. A. DE C. V. LIC. MARILU GARFIAS SANTOYO A QUIEN EN LO SUCESIVO SE LE DENÓMINARÁ LA EMPRESA Y POR LA OTRA EL SR.XXXXXXXXXXXXXXXXX XXXXX XXX A QUIEN EN LO SUCESIVO SE LE DENÓMINARÁ EL TRABAJADOR, AL TENOR DE LAS SIGUIENTES DECLARACIONES Y CLAUSULAS:

D E C L A R A C I O N E S

I. DECLARA LA EMPRESA:

a) Ser una Sociedad Anónima constituida conforme a las leyes de la República Mexicana, con domicilio en Insurgentes Sur No 3500 Col. Peña Pobre Código Postal 14060, en México, D.F.
b) Que tiene necesidad de contratar temporalmente los servicios del TRABAJADOR para que desempeñe la OBRA DETERMINADA consistente en la repartición de DIRECTORIOS TELEFÓNICOS con la categoría de REPARTIDOR AYUDANTE GENERAL, durante la distribución del directorio telefónico ZONAL PERISUR-COAPA Edición 2000 de la Ciudad de México

Continua declarando la EMPRESA que en la ejecución de la obra antes descrita, requiere de personal adicional a los trabajadores de planta y por lo tanto tiene la necesidad de contratar temporalmente al TRABAJADOR para que le preste servicios como REPARTIDOR AYUDANTE GENERAL, ya que dicha obra es de naturaleza extraordinaria y no puede desahogarse con el personal de planta.

II. DECLARA EL TRABAJADOR:

1. Que tiene los conocimientos necesarios para realizar la OBRA DETERMINADA especificada en el inciso b) de las declaraciones de la EMPRESA.

2. Que sus generales son los siguientes:

NACIONALIDAD: MEXICANA
SEXO: MASCULINO
ESTADO CIVIL: CASADO
EDAD: xx
No. Afiliación: xxxxxxxx-xx
No. R.F.C. xxxx-xxxxxx-xxx
DOMICILIO:

3. Que tiene la experiencia, conocimientos y aptitudes necesarias para llevar a cabo la labor de REPARTIDOR AYUDANTE GENERAL, requerida por la EMPRESA y que acepta su contratación por OBRA DETERMINADA en los términos de este contrato.

Una vez manifestadas las declaraciones anteriores, las partes están de acuerdo en otorgar las siguientes:

C L A U S U L A S

PRIMERA.- La EMPRESA contrata al TRABAJADOR para que preste sus labores con la categoría de REPARTIDOR AYUDANTE GENERAL, en la OBRA DETERMINADA d escrita en la declaración b) del personal contratado, sujetándose a la dirección, vigilancia e instrucciones que reciba de la EMPRESA por conducto de sus representantes.

SEGUNDA.- EL TRABAJADOR se obliga a desempeñar cualquier actividad conexa a su ocupación principal, siempre que se trate de trabajo contratado, cumpliendo para tal efecto con las instrucciones que reciba de los representantes de la EMPRESA facultados para ello, obligándose a atender todas las actividades conexas a su ocupación principal.

TERCERA.- EL TRABAJADOR conviene en que prestar sus labores en el lugar señalado en la declaración b) del presente contrato.

CUARTA.- Las partes convienen en que el presente contrato terminar el día: 7 de Abril del 2000 fecha en que se estima se habrá concluido la OBRA DETERMINADA objeto de la contratación, terminando en consecuencia, justificadamente la relación de trabajo sin responsabilidad alguna para las partes, en virtud del carácter temporal de los servicios prestados y de conformidad con la fracción III del articulo 53 de la Ley Federal de Trabajo.

QUINTA.- Las partes convienen en que la EMPRESA podrá rescindir sin su responsabilidad el presente contrato dentro de los primeros treinta días de prestar sus labores TRABAJADOR, si éste la engaña con certificados o referencias falsas en los que se atribuyan al TRABAJADOR capacidad, aptitudes o facultades requeridas para desempeñar el trabajo y de las cuales carezca.

SEXTA.- EL TRABAJADOR se obliga a desempeñar su trabajo con la intensidad, cuidado, puntualidad, intensidad y esmero adecuados, en el lugar y tiempo fijados para tal efecto a fin de ejecutar la OBRA DETERMINADA objeto del presente.

SÉPTIMA.- Las partes convienen en que el TRABAJADOR percibir por el trabajo prestado objeto de este contrato la cantidad de $ 1,500.00 (UN MIL QUINIENTOS PESOS 00/100 M. N.) como salario mensual, suma en que se incluye el pago de los séptimos días y el correspondiente a los días de descanso obligatorios. El pago ser hecho con moneda del curso legal y en dos exhibiciones los días 10 y 25 de cada mes en el domicilio de la EMPRESA, dentro de la jornada de trabajo o inmediatamente después de concluir esta.

OCTAVA.- EL TRABAJADOR se obliga a otorgar recibo en favor de la EMPRESA por la totalidad de los salarios ordinarios y extraordinarios devengados a que tenga derecho, conveniendose en que la firma implicar un finiquito total hasta la fecha del recibo correspondiente.

NOVENA.- EL TRABAJADOR prestar sus servicios bajo la dirección de la EMPRESA y queda entendido que por la naturaleza del trabajo a desarrollar, sus labores las desempeñara fuera de las instalaciones de la EMPRESA sin control de jornada de trabajo, obligándose a desahogar la misma bajo su mas estricta responsabilidad dentro de los limites establecidos en el Contrato Colectivo de Trabajo vigente en el centro de labores de 40 horas, por lo que las actividades a desempeñar no podrán considerarse como jornada extraordinaria, Queda expresamente convenido que el TRABAJADOR no laborará durante horas extraordinarias o durante días festivos de descanso obligatorio, si no es el consentimiento previo por escrito de la EMPRESA.

DECIMA.- EL TRABAJADOR disfrutar por cada cinco días de trabajo de dos días de descanso con goce de salario, los que se convienen por ambas partes serán el Sábado y Domingo de cada semana y cuyo salario queda ya pagado con la suma estipulada en la cláusula séptima de este contrato, por tratarse de salario mensual.

DECIMA PRIMERA.- EL TRABAJADOR al cumplir un año de servicios. disfrutar de un periodo de vacaciones pagadas en los términos del articulo 76 de la Ley Federal del Trabajo o la parte proporcional al tiempo laborado.

DÉCIMA SEGUNDA.- Las partes convienen en que el TRABAJADOR tendrá derecho a un aguinaldo anual que deber pagarse antes del 20 de diciembre de cada año cada equivalente a dieciocho días de salario.

DÉCIMA TERCERA.- EL TRABAJADOR tendrá además de las obligaciones indicadas en el presente contrato y las consignadas en el articulo 134 de la Ley Federal de Trabajo las que se enuncian a continuación:

A.- No divulgar ni dar a conocer en ningún tiempo, durante o después de la época de prestación de sus servicios, directa o indirectamente a cualquier persona o empresa, sin consentimiento previo y por escrito de la EMPRESA, ningún conocimiento o información que haya adquirido o adquiera durante el transcurso o con motivo de su trabajo, o ningún negocio en el cuál la EMPRESA haya estado o haya podido estar relacionada o interesada.

B.- Cuando por cualquier motivo se termine la relación de trabajo, deber entregar a la EMPRESA toda la documentación que en original o copia pertenezca o est, relacionado con negocios inherentes a la EMPRESA y que se encuentren en poder del EMPLEADO o est, bajo su control en forma directa o indirecta, así como los útiles de trabajo que le hayan sido otorgados.

DÉCIMA CUARTA.- EL TRABAJADOR se obliga a prestar sus labores en los términos del presente contrato y se obliga a respetar las disposiciones contenidas en el Reglamento Interior de Trabajo. La Ley Federal de Trabajo y demás ordenamientos legales o reglamentarios que le sean aplicables.

DÉCIMA QUINTA.- Ambas partes se obligan a cumplir con los planes y programas de capacitación y adiestramiento establecidos en los centros de labores, de acuerdo a lo estipulado en la Ley Federal de Trabajo.

DÉCIMA SEXTA.- Ambas partes se obligan a someterse a las disposiciones que en materia de seguridad social establece la Ley del Seguro Social y demás preceptos relativos y aplicables de la Ley Federal de Trabajo.

DÉCIMA SÉPTIMA.- Para seguridad de los contratantes, el TRABAJADOR estar obligado a someterse a los exámenes médicos que acuerde el patrón y a poner en practica las medidas profilácticas que el mismo o las autoridades del ramo acuerden.

Leido que fue el presente contrato, las partes lo firman de conformidad y por duplicado en la Ciudad de México,

LA EMPRESA

LIC. MARILU GARFIAS SANTOYO
JEFE DE PERSONAL

EL TRABAJADOR

xxxxxxxxx xxxxxxxx xxxxxxxx

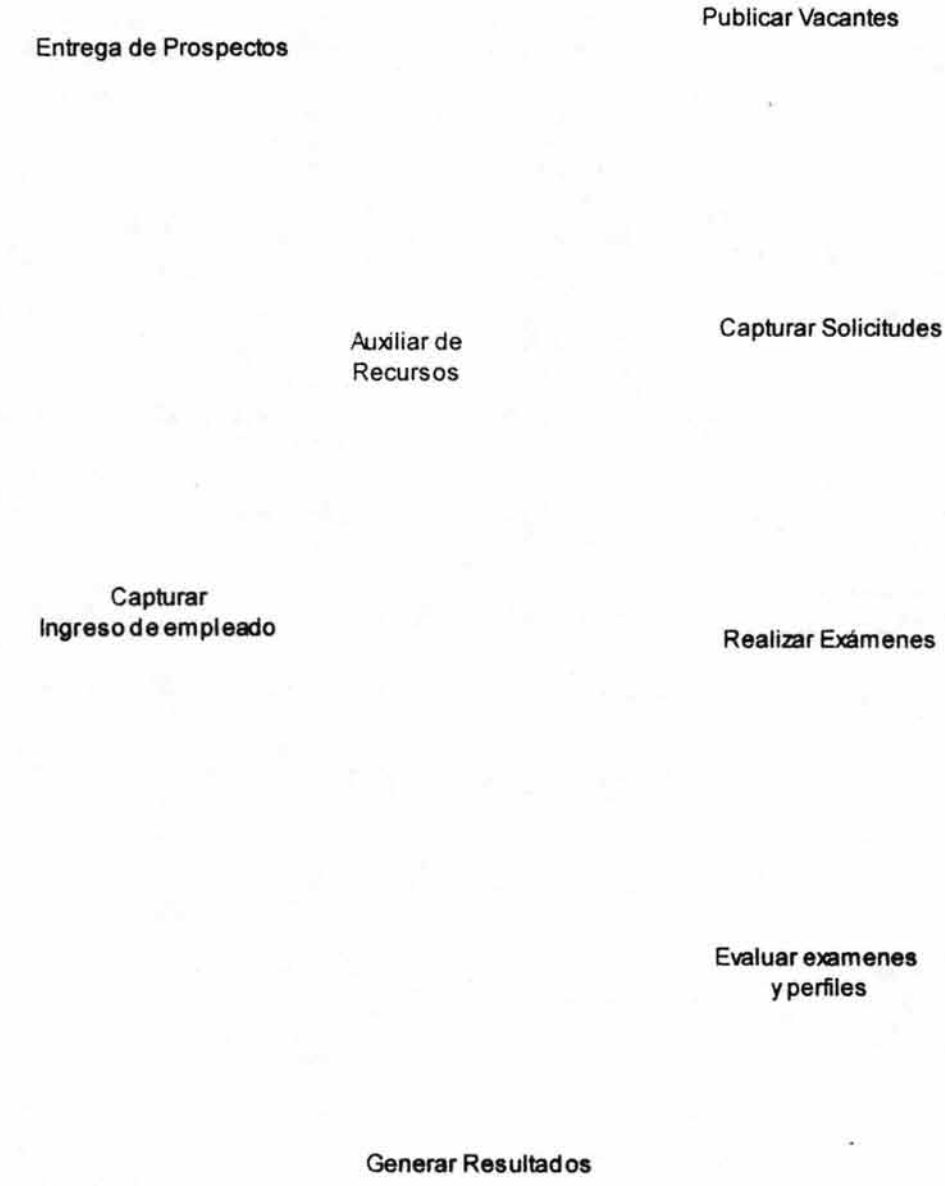


Figura 2.1
Caso de uso de Reclutamiento y Selección

2.7.2 Análisis del Módulo de Estructura Organizacional

La estructura Organizacional en cuanto a niveles y puestos, la podemos ver en la siguiente tabla:

Nivel	Puesto	Descripción
1	1	REPARTIDOR AYUDANTE GENERAL
1	2	AUXILIAR DE JEFE DE DISTRIBUCION
1	3	AUXILIAR DE BODEGA
1	4	AUXILIAR DE ALMACEN
1	5	VIGILANTES
1	6	CHOFER
1	7	OPERADOR DE MONTACARGAS
8	100	COORDINADOR DISTRIBUCION
10	101	SECRETARIA DE SUBDIRECCION
11	102	SECRETARIA DE DIRECCION
12	103	AUXILIAR DE CAPACITACION
13	104	CAJERO
14	105	ANALISTA DE COMUNICACION
14	106	ANALISTA DE FACTURACION
14	107	ANALISTA DE MET Y PROCED
14	108	ANALISTA DE PRODUCCION Y LOGISTICA
14	109	ANALISTA DE PROCESOS Y COMUNIC.
14	110	AUXILIAR DE PERSONAL
14	111	AUXILIAR DE RECLUTAMIENO Y SELECCION
14	112	COORDINADOR ASIGNACIONES
14	113	COORDINADOR COSTOS
14	114	COORDINADOR RECURSOS HUMANOS
15	115	ADMDOR DE RED
15	116	ADMDOR DE SISTEMAS
15	117	ADMDOR DE SISTEMAS VAX
15	118	ADMOR SISTEMAS UNIX
15	119	ANALISTA PROGRAMADOR
15	120	COORDINADOR EDICION
15	121	COORDINADOR FORMACION
15	122	COORDINADOR TECNICO
15	123	EDITOR
15	124	ENCARGADO DE PRESUPUESTOS
15	125	GERENTE TECNOLOGIA DE LA INFORM.
15	126	SUPERVISOR MERCADOTECNIA PAG WEB. ECO
16	127	SUPERVISOR DE SERVICIO A CLIENTES
16	128	SUPERVISOR DE TELEMARKETING
17	129	ASPIRANTE SUPERVISOR. VTAS
17	130	CONTADOR DE INGRESOS
17	131	CONTADOR EGRESOS
17	132	COORDINADOR CAPACITACION
17	133	GERENTE PUBLICIDAD PAG AMARILLAS Y

17	134	SUBGERENTE ADMINISTRATIVO
17	135	SUPERVISOR DE VENTAS ZONALES
17	136	SUPERVISOR DE VENTAS
17	137	SUPERVISOR DE VENTAS INTERNAC.
18	138	ABOGADO
18	139	CONTADOR DIVS FORAN. Y FISCAL
18	140	COORDINADOR REGIONAL METRO
18	141	COORDINADOR REGIONAL NORTE
18	142	COORDINADOR REGIONAL SUR
18	143	JEFE DE COMPRAS
18	144	JEFE DE CONTROL PRESUPERVISORUESTAL
18	145	JEFE DE GESTION DE CARTERA
18	146	JEFE DE GESTION LEGAL
18	147	LIDER ANALISIS Y SOPORTE USUAR
18	148	LIDER ANUNCIOS ESPECIALES
18	149	LIDER ASIG. E INGRESOS DATOS
18	150	LIDER CONTROL PRODUCCION
18	151	LIDER DESARROLLO
18	152	LIDER GRAFICOS
18	153	LIDER INFRAESTRUCTURA
18	154	LIDER INGRESO DE DATOS
18	155	LIDER ORDEN DE SERVICIO
18	156	LIDER POT
18	157	LIDER PRUEBAS
18	158	LIDER SISTEMAS
18	159	LIDER WKO
19	160	GERENTE DE INTEGRACION Y DESARROLLO
19	161	GERENTE DE FOROS
19	162	GERENTE DE METODOS Y PROCEDIMI
19	163	JEFE DE FACTURACION
19	164	JEFE DE PERSONAL
20	165	GERENTE DE ALIANZAS Y NVOS NEG
20	166	GERENTE DE SERVICIO A CLIENTES
20	167	GERENTE DIVISIONAL II
20	168	GERENTE MERCADOTECNIA INTERNET
20	169	GERENTE TELEMARKETING
21	170	CONTRALOR
21	171	GERENTE DE DISEÑO GRAF Y PRODUCCCION
21	172	GERENTE DE DISTRIBUCION
21	173	GERENTE DE INFRAESTRUCTURA
21	174	GERENTE DE PLANEACION Y CONTROL PTO
21	175	GERENTE DE PROCESOS DE INFORMACION
21	176	GERENTE DE SISTEMAS DE PRODUCCION
21	177	GERENTE DE SISTEMAS INFORMACION
21	178	GERENTE DIVISIONAL I
23	179	GERENTE DE TECNOLOGIA
23	180	GERENTE REGIONAL
24	181	SUBDIRECTOR DE REC HUM Y LABORALES

25	182	SUBDIRECTOR DE SISTEMAS
25	183	SUBDIRECTOR TECNICO
26	184	DIRECTOR ADMINISTRATIVO
26	185	DIRECTOR E-BUSINESS
27	186	SUBDIRECTOR GENERAL DE VENTAS

2.7.3Análisis del Módulo de Capacitación

Esta entidad pertenece al Departamento de Recursos Humanos, esta se encarga de gestionar absolutamente toda la capacitación de todos los empleados de Sección Amarilla, consta de los siguientes Apartados:

- Cursos Internos
- Cursos Externos
- Gestión de Aulas
- Equipo, material
- Organización y Desarrollo

Cursos Internos

Estos cursos son los que se imparten de y para la empresa, es decir, dentro de la empresa existen cada vez mas y mejores procedimientos para su desarrollo, estos procedimientos son creados por los mismos empleados, por unos o por varios, una vez que estos procedimientos se encuentran bien definidos en un documento, se toma la decisión en base a un estudio de impartir un curso sobre estos mismos, o bien son solicitados por los mismos empleados para mejorar su rendimiento, en cualquiera de los dos casos el curso se planea.

Cursos Externos

Son aquellos que son impartidos por empresas ajenas a Sección Amarilla, estos cursos se pueden impartir a causa de una nueva necesidad de conocimiento, el cual no lo posee ninguno de los empleados.

Gestión de Aulas

No es mas que el control de asignación de aulas para los cursos a impartir. Las aulas no solo son utilizadas para cursos, también se utilizan para diferentes eventos, o bien pueden ser solicitadas a Capacitación.

Equipo material

Para lo Cursos y eventos que existen, se cuenta con equipo material, como monitores, PC's, Cañón electrónico, pizarrón electrónico, etc.

Organización y desarrollo

Aquí se gestionan, planean, administran todas las entidades antes mencionadas.

La toma de decisión para que se imparta un curso, se hace en base a diferentes situaciones:

- Fue solicitado
- Se detectó la necesidad

Una vez que ya es un hecho que se va a impartir el curso, se toman en cuenta los siguientes datos:

Curso = Nombre del Curso +
Fecha de inicio +
Fecha de termino +
Horario +
Alumno +
Evaluación parcial por Alumno +
Evaluación total por Alumno +
Evaluación Total por Grupo +
Profesor o Empresa que lo imparte+
Lugar donde se va a impartir +

Horario = Días de la semana +
Hora de inicio +
Hora de término

Alumno = Número de empleado +
Nombre +
Apellido paterno +
Apellido materno

Profesor = Si es empleado o no +
Nombre del empleado o Empresa +
Si es empresa, Nombre del representante +

Institución = Nombre +
Dirección

Costo del Curso = Cantidad +
Fecha de pago +

Equipo = Número de serie +
Nombre del Equipo

Aula = Número de Aula
Nombre del Aula

Préstamo del

Equipo y Aula = Fecha de préstamo +
Hora de préstamo +
Fecha de entrega +
Hora de entrega +
Nombre de la persona que solicitó el préstamo

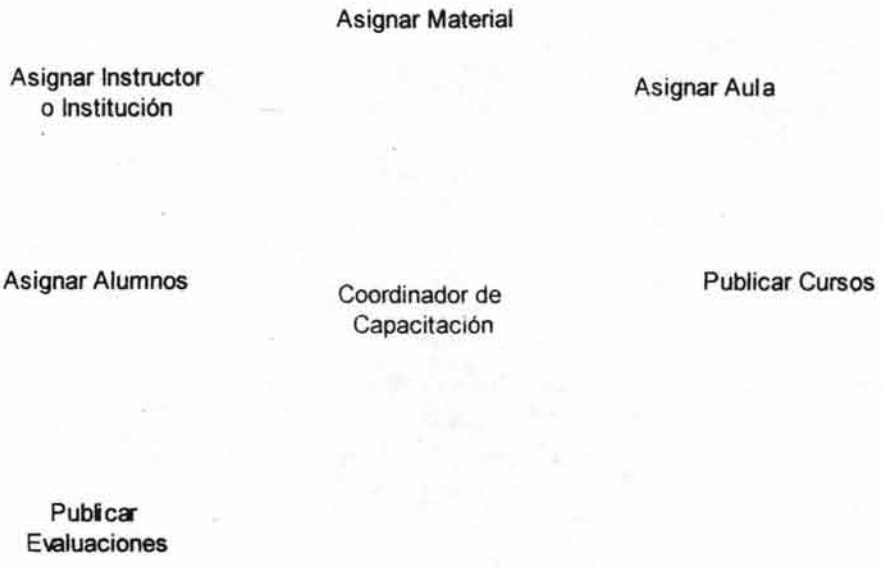


Figura 2.2
Caso de uso de Capacitación

2.7.4 Análisis del Módulo de Nómina

El módulo de Nómina está constituido por 3 nóminas principales, que contienen a todos los empleados de la Sección Amarilla:

- Nómina de Agentes de Ventas
  - Contrato Anterior
  - Contrato Nuevo

- Nómina del Personal Administrativo
  - Personal de confianza ingreso al 1 de junio 1991
  - Personal de confianza ingreso al 2 de junio 1991
  - Personal sindicalizado ingreso al 1 de junio 1991
  - Personal sindicalizado ingreso al 2 de junio 1991

- Personal de Reparto

Nómina comprende los siguientes módulos:

- IMSS
  - Cálculo mensual de salarios mixtos(Nómina del personal Administrativo)
  - Cálculo de salarios Integrados por cambio de factor(Nóminas Administrativos y Reparto).
  - Cálculo mensual de salario variables (Nómina de Agentes de Ventas).
  - Impresión de listados de los movimientos del IMSS
  - Histórico de sueldos y puestos
  - Listado de transferencia ASCII a SUA
  - Histórico de movimientos IMSS

- Cálculo de Nómina
  - Fórmulas
    - Mantenimiento
    - Copiado
    - Asignación por tipo de empleado
  - Incidencias
    - Asignación
    - Mantenimiento
  - Generación de la nómina
    - Cálculo de todas las fórmulas establecidas
    - Cálculo del impuesto mensual o quincenal de acuerdo al artículo 80
    - Cálculo del impuesto de mensual o quincenal acuerdo al artículo 86
    - Cálculo del impuesto Anual
    - Cálculo de formulas mas elaboradas
  - Listado de movimientos
  - Bitácora de cálculo
  - Mantenimiento de movimientos
  - Impresión de la nómina
  - Impresión de recibos
  - Relación de depósitos y cheques

- Layout de depósitos
- Control de saldos deudores
- Cálculo de sábanas de percepciones con y sin prestaciones para actualizar salarios integrados en Agentes de ventas
- Generación de nóminas especiales
- Póliza contable
- Actualización de acumulados
- Impresión de cheques

- Finiquitos
  - Mantenimiento
  - Cálculo
  - Cálculo de liquidación
  - Histórico de bajas del empleado

- Procesos especiales
  - Aumento de salarios
  - Reparto de utilidades

- Fondo de Ahorro
  - Cálculo de préstamos
  - Reportes
    - Layout sindicalizados
    - Layout confianza
    - Layout préstamos Inbursa
    - Layout préstamos Bital
    - Reporte de Saldos
    - Prestamos pagados por adelantado
    - Liquidación del fondo de ahorro

Una nómina está compuesta por:

- Periodo(s)
- Empleado(s)
- Movimientos Actuales
- Movimientos Acumulados o Periodos Acumulados

Una nómina es la que contiene todos los cálculos de los empleados en un cierto periodo de tiempo, el cual es fijado con anterioridad, la nómina puede contener desde 1 hasta n periodos, desde 1 hasta n empleados y desde 0 hasta n movimientos por empleado. Una nómina puede contener el cálculo de varios periodos,

Existen dos tipos de nómina:

Nómina Normal.- Este tipo de nómina se refiere a la nómina base, es decir, la que contiene los empleados originales. En esta nómina se calculan los periodos normales, ningún caso en especial, se pueden encontrar los 24 periodos, es decir las 24 quincenas del año. Por lo tanto a estas nóminas se les conoce como principales.

Nómina Especial.- Son aquellas que se utilizan para presentar nóminas complementarias en el caso de algún error, pago de Finiquitos, Pago de utilidades, de aguinaldo, etc.

Periodo.- Un periodo es aquel que se crea en una nómina existente, este comprende un intervalo de fechas sin número determinado de días. Un periodo normalmente es una quincena calculada de la nómina, o bien unos cuántos días o incluso un día en el caso de nóminas especiales.

Periodo Acumulado.- Se refiere a todos los movimientos acumulados de un determinado periodo.

Movimientos.- No son mas que las percepciones y deducciones ya calculadas, así como los cálculos estadísticos o informativos, los cuales no aparecen en la impresión de la nómina, ni en los recibos.

Movimientos Actuales y Acumulados.- Los movimientos tienen un estatus, Actuales ; son los últimos movimientos calculados, Acumulados; son los movimientos de periodos anteriores, los cuales son requeridos cuando un cálculo requiere de un saldo anterior.

2.7.4.1 Conceptos

Conceptos de la nómina de Agentes de Ventas

Concepto.- Es el nombre que se le da a cada formula, cada nómina tiene sus propias claves de concepto:

Clave	Concepto
102	Acum Ayuda Despensa
908	Acumulación De Premios
31	Acumulado Del Fondo De Ahorro
99	Acumulado Fondo Ahorro
906	Adeudos División
30	Aportación Fondo Ahorro Empleado
29	Aportación Fondo Ahorro Empresa
103	Aportación Fondo Ahorro Empresa D
902	Aportación Voluntaria
59	Ayuda De Despensa
60	Ayuda De Renta
888	Bonificaciones A Clientes
907	Bono Prod Vtas
12	Cancelaciones
7	Comisiones
904	Comisiones A Bajas
903	Comisiones Pendientes Bajas
19	Compensaciones
556	Crédito Al Salario
900	Cuota De Capacitación
64	Cuota Sindical
815	Diferenciales De Comision
817	Dscto Pmo Ac Df
37	Fonacot
GAS	Importe Mensual De Gas
42	Impuesto
38	Infonavit
34	Intereses Prestamo Fdo Ahorro
41	Pension Alimenticia
39	Prestamo Compania
819	Prestamo Df
826	Prestamo Df 2000
32	Prestamo Fondo De Ahorro
820	Prestamos A Comis Cta 99
2	Prima Vacacional
35	Seguro De Autos
36	Seguro De Gastos Medicos
9	Traspasos
195	Uniformes
3	Vacaciones

Conceptos de la nómina del Personal Administrativo

Clave	Concepto
@CSC	Credito Al Salario Calculado
@CSP	Credito Al Salario Pagado
@GRA	Gratificacion Anual
@HEX	Horas Extras Dobles
@ISP	Es Un Impuesto
@PVA	Prima Vacacional
@VAC	Pago De Vacaciones
ACFA	Acumulado Fondo Ahorro
HACIA	Adeudo Viaticos
ADPI	Ajuste De Dias Por Ingreso
AFEM	Aportacion Fondo Ahorro Empresa
AFON	Aportacion Fondo Ahorro Empresa
ANTIG	Antiguedad
APFO	Aportacion Fondo Ahorro Empleado
AREN	Ayuda De Renta
BONO	Bono
CAST	Castigo
CCUR	Compensacion De Curso
COME	Es Una Compensacion
COMP	Compensacion
CURS	Descuento Curso Ingles
DCOM	Deduccion De Pcome
DESP	Ayuda De Despensa
DIFA	Diferencia Por Aumento
DINF	Diferencias Infornavit
FALT	Faltas
FONA	Fonacot
FVAC	Vacaciones Pagadas Finiquito
HRTR	Horas Extras Triples
INCA	Incapacidad
INCM	Incapacidad Por Maternindad
INCR	Incapacidad Por Riesgo De Trabajo
INFO	Pago Infonavit
INTE	Intereses
MAQU	Descuento De Computadoras
MAUS	Mausoleos
PALI	Pension Alimenticia
PANT	Prima De Antiguedad
PCIA	Prestamo Compania
PCOM	Percepcion Compensacion
PDOM	Prima Dominical
PESG	Permiso Sin Goce De Sueldo
PFIN	Pago De Finiquitos
PFOA	Descuento Pres Fdo Ahor
PLIQ	Pago De Liquidacion
PNOR	Percepcion Normal
PPVA	Pago Prima Vacacional Dias No Disf
PRES	Prestamo
PVAC	Pago De Vacaciones
RETR	Retroactivo

RVOL	Retiro Voluntario
SEAU	Seguro De Automovil
SGMM	Seguro Gastos Medicos
SINC	Subsidio Incapacidad
SOSU	Sobre Sueldo
TRAN	Ayuda De Trasporte
UNIF	Uniformes

Conceptos de la Nómina de Repartidores

Clave	Concepto
1	Percepcion Normal
14	Horas Extras Dobles
15	Horas Extras Triples
19	Compensaciones
24	Prima Dominical
41	Pension Alimenticia
42	Impuesto
556	Credito Al Salario
59	Ayuda De Despensa
64	Cuota Sindical
DIFA	Diferencia De Sueldo
DIFE	Diferencia En Sueldo
INFO	Pago Infonavit
RETR	Retroactivo

Salario Diario Percepción Normal.- Este salario es la base de ingresos neta para todos los cálculos del empleado. Esta cantidad cambia cuando existen cambios de sueldos, aumentos, etc. En el caso de Agentes de Ventas esta cantidad es promedio, ya que en esta nómina el SDPN se basa en las comisiones, por lo tanto todo el tiempo será variable.

Salario Diario Integrado.- Este salario es el que se toma en cuenta para el cálculo de la cuota del IMSS. Y se calcula de la siguiente manera:

Para el cálculo del Salario Integrado de la nómina de Agentes de Ventas, es necesario calcular antes el siguiente factor:

Ayuda de despensa = Suma de movimientos del concepto 59
Fondo de Ahorro Empresa = Suma de movimientos del concepto 29
Conceptos IMSS, SAR, INFONAVIT = Suma de movimientos diferentes a 29 y 59

Factor = (Salario Mínimo * 0.4) * 30.4

Si la ayuda de despensa es mayor al factor calculado, entonces, se calcula el excedente

Excedente = Ayuda de despensa - Factor

El 3% de la Aportación Empresa = (Fondo de Ahorro Empresa / 0.13) * 0.03

Salario Integrado Agentes de Ventas = (El 3% de la Aportación Empresa + Excedente + Conceptos IMSS, SAR, INFONAVIT)

El tope para dicho salario es de 25 salarios minimos

Para el cálculo del SDI INFONAVIT y SAR , es el mismo procedimiento, solo que los topes son diferentes:

Tope INFONAVIT = 16 salarios mínimos
Tope SAR = 25 salarios mínimos

Para la nómina de reparto

Como todos los empleados son eventuales, por lo tanto se toman en cuenta la siguientes prestaciones, las cuales pueden cambiar durante el año, o hasta el año siguiente:

- 25 días de Aguinaldo
- 14 días de gastos educacionales
- 60% prima vacacional
- 5 días de vacaciones no disfrutables
- 30% de prima de vacaciones sobre los 5 días no disfrutables
- Vacaciones disfrutables por ley
- 3% de ayuda de despensa
- $175 al mes de ayuda de transporte

Se calculan los factores correspondientes:

Para 1 año laborado

Aguinaldo	25/365	0.0685
Gastos educacionales	14/365	0.0384
Prima vacacional	(6 * 0.6 + 5 * 0.3) / 365	0.0140
Vacaciones no disfrutables	5/365	0.0137

Sueldo 1.00

1.1345

Para 2 años laborados

Aguinaldo	25/365	0.0685
Gastos educacionales	14/365	0.0384
Prima vacacional	(8 * 0.6 + 5 * 0.3) / 365	0.0173
Vacaciones no disfrutables	5/365	0.0137

Sueldo 1.00

1.1378

Entonces si un repartidor trabaja 1 año y su sueldo diario = 61.6

Su salario integrado = (61.60 * 1.1345) + Transporte diario + Despensa diaria

Su salario integrado = (61.60 * 1.1345) + 5.83 + 1.848 = 77.5632

Y para la Nómina de Administrativos

Primero el Salario Diario Integrado Variable (SDIV):

SDIV = (Suma de los movimientos: @HEX,@HTRT,COMP,PDOM,BONO) / (Número de días del mes a calcular)

Para el cálculo es necesario tener factores establecidos:

Factores aplicables para la determinación del salario diario integrado al 2 de Junio de 1999

PERSONAL DE CONFIANZA INGRESO AL 1 DE JUNIO 1991

Intervalo Antiguedad(anos)		I.M.S.S.	Infonavit
1	4	1.2887	1.2887
5	9	1.295	1.295
10	14	1.3014	1.3014
15	19	1.3078	1.3078
20	24	1.3141	1.3141
25	29	1.3205	1.3205
30	34	1.3268	1.3268
35	39	1.3332	1.3332

PERSONAL DE CONFIANZA INGRESO DEL 2 DE JUNIO 1991

Intervalo Antiguedad(anos)		I.M.S.S.	Infonavit
1	1	1.1563	1.1563
2	2	1.1596	1.1596
3	3	1.1629	1.1629
4	4	1.1662	1.1662
5	9	1.1695	1.1695

PERSONAL SINDICALIZADO INGRESO 1 JUNIO 1991

Intervalo Antiguedad(anos)		I.M.S.S.	Infonavit
1	4	1.2887	1.2887
5	9	1.295	1.295
10	14	1.3014	1.3014
15	19	1.3078	1.3078
20	24	1.3141	1.3141
25	29	1.3205	1.3205
30	34	1.3268	1.3268
35	39	1.3332	1.3332

PERSONAL SINDICALIZADO INGRESO 2 DE JUNIO 1999

Intervalo Antigüedad(anos)		I.M.S.S.	infonavit
1	1	1.1563	1.1563
2	2	1.1596	1.1596
3	3	1.1629	1.1629
4	4	1.1662	1.1662
5	9	1.1695	1.1695

Si el empleado no ha cumplido un año, se toma 1 año de antigüedad.

Salario Integrado IMSS y SAR
Si el empleado es eventual, se resta a cada factor, el fondo de ahorro que se traduce en un 3%
A estos factores se le deberá acumular:
Ayuda de transporte = $5.00 diarios
Ayuda de renta = $1.70 diarios (Solo contrato anterior)
Excedente ayuda de despensa = $13.78 diarios
Se contempla un tope:
Tope = (Salario Mínimo) * 25
Por lo Tanto:

SDI = SDPN * (FACTOR)

SDI = SDI + 1.70 + 5.00 + (Excedente del 40% del salario mínimo) + SDIV

Si SDI es mayor al tope: SDI = Tope

Salario Integrado INFONAVIT

Se contempla un tope:
Tope = (Salario Mínimo) * 16
Por lo tanto
SDI = SDPN * (FACTOR)
Salario Diario Integrado Administrativos = SDI + 1.70+ (Excedente del 40% del salario mínimo) + SDIV
Si SDI es mayor al tope: SDI = Tope

Salario Real
SDI = SDPN * (FACTOR) + 1.70+ (Excedente del 40% del salario mínimo)

2.7.4.2 Cálculo de la nómina

Fórmula

Por cada concepto existe una formula, con el mismo nombre del concepto, es decir, cuando se desea pagar un concepto, se calcula por medio de la formula, la cuál necesita el nombre del concepto para poder hacer el cálculo, así como también la incidencia por empleado. Entonces, si la formula se encarga del cálculo, el resultado de ella, es un movimiento, de tal manera que la relación con los demás es:

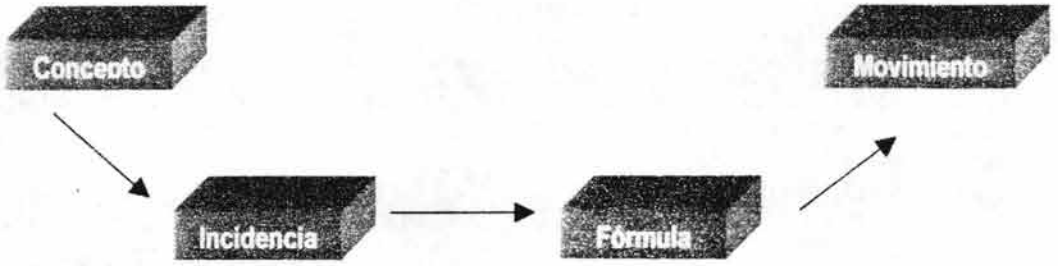


Figura 2.3
Flujo de un concepto en la nómina

Copiado de fórmulas.- Cuándo se crea una nueva nómina, es común querer tener todas las fórmulas existentes en otra nómina, mas las que se van a crear en esta nueva nómina, por eso es posible copiar todas las formulas o solo algunas con todos sus parámetros y especificaciones.

Formulas por tipo de empleado.- Cada tipo de empleado tiene unas formulas asignadas:
Nómina del personal Administrativo. Las siguientes fórmulas no se aplican a todo el personal eventual:

AFON.- APORTACION FONDO AHORRO EMPRESA
APFO .-APORTACION FONDO AHORRO EMPLEADO
AFEM.-.APORTACION FONDO AHORRO EMPRESA
ACFA.- ACUMULADO FONDO AHORRO

Nómina de Agentes de Ventas. Las siguientes fórmulas no se aplican a todo el personal eventual:

29 APORTACION FONDO AHORRO EMPRESA

30 APORTACION FONDO AHORRO EMPLEADO

103 APORTACION FONDO AHORRO EMPRESA D

Nómina del personal de reparto . Las siguientes fórmulas no se aplican a todo el personal eventual

41 PENSION ALIMENTICIA

Componentes de una formula

Secuencia.- Indica el orden en qué se calcularán las formulas, ya que muchas de ellas se componen de otras formulas, por lo tanto es importante llevar una secuencia de cálculo.

Clave .- Mismo nombre del concepto de hasta 4 posiciones alfanuméricas de longitud variable.

Tipo de fórmula.- Puede ser: Percepción, Deducción o Estadística (sólo informativa).
Base I.S.P.T.- Se refiere a la base gravable, puede ser:

Base	Descripción
GE	Gravable Extraordinaria aplicación del articulo 86, se acumula en @PGE
EX	Exento totalmente, se acumula en @EXE
NI	Ni Gravable ni exento. (ejemplos: viáticos, TRAN)
EE	Exento con tope (lo que excede el tope se grava como extraordinario, aplicando el articulo 86, se acumula en @EXE y @PGE.

Dónde:

@TPG. Total de Percepción Gravable, es decir, los movimientos que generen @TPG, son la base gravable neta, sobre la cuál se calculará el impuesto.

@PGE. Percepción Gravable Extraordinaria, esta es la base para el cálculo del impuesto según el artículo 86

@EXE. Exento, cantidad totalmente exenta de impuesto, es decir, el movimientos que genere un @EXE, su cantidad estará intacta y no se tomará en cuenta para el cálculo del impuesto.

Tope no gravable.- El resultado de la formula, tendrá un tope, llegando a este, el resto del monto será gravable
Tope mensual.- Se topa el resultado de la formula en el periodo par.
Recibo.-Indica si el movimiento generado se va a imprimir en el recibo.
Secuencia en la prioridad de descuentos.- Indica si esta formula se incluirá en la prioridad de descuentos.

Operadores

- + Suma
- - Resta
- * Producto
- / Cociente
- = Igualación

Operándos.- Los operándos de una fórmula pueden ser:

- Constante. Cantidad fija.

  Formato: |nnnn.nnnn o |dd/mm/aa
  Dónde

  | = Indica que es una constante
  nnnn.nnnn = Es la cantidad.

  Ejemplo:
  |360
  La constante es 360.
  |30/01/01
  La constante es 30/01/01.

- Fórmula. Resultado de otra fórmula con secuencia anterior.
- Movimientos acumulados de la misma nómina o de otras, especificando periodos, si es importe (cantidad monetaria) o dato(número de días).

  Formato @ccccidddnnnaa

  Dónde:

@ = Indica qué el operando será un movimiento acumulado.
Cccc = nombre del concepto.
I = Dígito que nos indica qué se obtendrá del acumulado, si es importe = 0, o dato = 1.
ddd = Nos indican de qué periodo se obtendrá el acumulado.
nnn = Sí no se omite este parámetro, quiere decir que se requiere todo el acumulado de un intervalo de periodos, siendo el inicial 'ddd' y el final 'nnn'.(opcional)
aa = Indica de qué año será el acumulado.(opcional)

Ejemplo:

@PNOR0004

Se requiere el acumulado del concepto PNOR(Percepción Normal para la nómina del personal Administrativo), se requiere su importe del periodo 4 del año en curso.

@00071001024

En este ejemplo se requiere el acumulado del concepto 7(Comisiones de la nómina de Agentes de Ventas), se requiere el dato del periodo 1 hasta el periodo 24 del año en curso.

@001001012100

En este ejemplo se requiere el acumulado del concepto 1(Percepción Normal de la nómina de Repartidores), se requiere importe del periodo 10 al 121 del año 2000

- Datos acumulados pertenecientes al diccionario de datos

Formato: @/ccccidddnnnaa

Ejemplo:

@/SDPN000101000
Se requiere el acumulado del Sueldo Diario Percepción Normal, del periodo 1 al periodo 10 del año 2000. En este caso el parámetro i, debe ser siempre 0.

- Diccionario de datos. Se refiere a la información ya existente, es decir se hace referencia a los datos correspondientes

Formato: /cccc
Dónde:

'/ '= Indica que es un dato del diccionario de datos.
cccc= La clave del dato

- **SDI** = Salario Diario Integrado.
- **SDPN** = Salario Diario Percepción Normal.
- **SM** = Salario mínimo.
- **SSAR** = Salario Diario Integrado base SAR.
- **SBI** = Salario Diario base INFONAVIT.
- **IMP** = Importe de la incidencia insertada en la Asignación de incidencias.
- **DAT** = Dato de la incidencia insertada en la Asignación de incidencias.
- **FING** = Fecha de ingreso
- **FBAJ** = Fecha de baja
- **FHOY** = Fecha de hoy

Ejemplo:

/SDPN
Se desea obtener el Salario Diario Percepción Normal actual.

Especificaciones de una fórmula

Tope.- Tope del resultado de la fórmula, el cual puede ser especificado por:
'C' =Constante.
'S' =Número de salarios mínimos.
'D' =La clave de una fórmula calculada en el mismo periodo, con secuencia inferior.
Ejemplo

C3000 =El tope es la constante 3000
S10 =El tope es 10 salarios mínimos
DPNOR=El tope es el resultado de la fórmula PNOR.

El proceso de cálculo compara el resultado de la fórmula con el tope establecido, de tal manera que, el resultado debe de ser menor igual al tope.

Saldos.-Control de saldos deudores, el activar esta opción significa que los saldos del movimiento se imprimirán en los recibos.

Redondeo.-El redondeo que se aplica al importe o dato del resultado de la fórmula, siguiendo los siguientes criterios:

0 = No aplica redondeo, la cantidad queda como se calculó: 1,453,264.72
1 = Redondeo a la unidad: 1,453,265
2 = Truncar a la unidad: 1,453,264
3 = Subir a la siguiente unidad, sin importar los decimales que se tengan: 1,453,265

Base 2%.- El resultado de la fórmula * 0.02, este dato servirá para la realización de las pólizas.
Incidencia.- Indica si esta fórmula requiere incidencia.
Préstamo.- Saldo inicial de la fórmula
Saldo insoluto.- Préstamo – descuentos
Factor PTU.- Último factor obtenido del reparto de utilidades.
Importe PTU.- Último importe obtenido del reparto de utilidades.

Incidencias.- Son los Valores base que la formula se encarga de utilizarlos, estos valores son calculados previamente por el personal administrativo y pueden ser valores numéricos que se traducen en cantidades monetarias como bonos o descuentos especiales, o bien en cantidades enteras como horas extras, días extras, etc.

Incidencias fijas.- Son las incidencias, que una vez asignadas, siempre se aplicarán al empleado, sin tener que asignarlas en cada periodo, como se hacen con las incidencias normales.

Asignación de Incidencias.- La asignación de incidencias puede ser; individual, por concepto, a todos los empleados, o a todo un centro de costo, o bien, cuándo es una gran cantidad de incidencias, se importan por medio de un archivo.
Una incidencia puede ser un valor directo como un BONO = 20,000.00, en este caso la incidencia se vería de la siguiente manera:

Archivo **Incidencia.csv** a importar.

Empleado	Concepto	Importe	Dato
2356	BONO	20000	0

Nótese que el importe no lleva comas.

En el siguiente ejemplo de incidencia, a diferencia del anterior ahora se van asignar días, es decir un valor en la columna dato; INCA = 5, el concepto es INCAPACIDAD, esto quiere decir que se asignarán al empleado 5 días de incapacidad, y en vez de cobrar 15 días de salario cobrará solo10, este cálculo se realiza por medio de la formula INCA.

Empleado	Concepto	Importe	Dato
2356	INCA	0	5

Generación de la nómina

Todas las formulas creadas, junto con todos los parámetros elegidos, ya mencionados anteriormente, son integrados en este proceso, dónde se hace efectivo el cálculo de la nómina, guardando todos los resultados y asignándoles el nombre del concepto correspondiente. Antes de generar la nómina, se verifica a qué empleados se les calculará impuesto en el primer periodo, ya que en el segundo periodo a todos se les calcula impuesto. Se genera por tipos de nómina, y es posible generar desde 1 hasta todos los empleados.

Prioridad de descuentos

Este proceso se realiza durante la generación de la nómina, y se aplica a la nómina de Agentes de Ventas y personal Administrativo.

En prioridad de descuentos se incluyen solo aquellas formulas que eligió el usuario en el momento de cargarlas, es decir, al crear una formula se elige si esta estará incluida en una prioridad de descuentos, si es así, durante la generación de la nómina y después de haber calculado absolutamente todos los descuentos, se calcula un tope, se realiza una sumatoria de los descuentos hasta que se llegue al tope calculado, el resto de los descuentos se mantienen latentes para la siguiente quincena, estos no se aplicarán hasta que sean manipulados en el siguiente cálculo de prioridad de descuentos, sin embargo en la quincena actual que no se pudieron aplicar, se mantendrán como informativos.

Se requiere en cierto momento de agregar conceptos a la prioridad de descuentos, o bien, modificar las prioridades de los conceptos existentes.

Nómina del Personal Administrativo

SDPN = Salario Diario Percepción Normal
SALMIN = Salario mínimo
TOPE = [(SDPN * Días Trabajados) + (otras percepciones COMP, BONO, CCUR, @PVA, @HEX, HRTR, RETR,) – SALMIN * Días Trabajados] * 0.3

Fórmulas y secuencia de descuentos:

Fórmula	Prioridad
INTE	1
DINF	2
PCIA	3
HACIA	4
AREP	5
CURS	6
UNIF	7
MAQU	8
SEAU	9
SGMM	10
RVOL	11
PFOA	12

Nómina de Agentes de Ventas
7 = Comisión.
9 y 10 = Traspasos, siempre que éstos sean positivos.

TOPE = [(7 + 10 + 9 + 8 + 8A) – (SALMIN * 15)] * 0.3

Fórmula	Prioridad
815	1
32	2
35	3
906	4
826	5
9	6
10	7
39	8
36	9
37	10
888	11
12	12
820	13
195	14
32	15
902	16

Ejemplo con un empleado de la nómina de Agente de ventas.

Empleado	Concepto	Tipo movimiento	Monto
890078	7	P	3900
890078	35	D	3000
890078	36	D	1700
890078	39	D	6000
890078	195	D	1500

TOPE = (3900 – 632.25) * 0.3 = 980.32

Según la prioridad, el concepto 35 tiene mayor prioridad, por lo tanto es el primero que se intenta descontar:

SI 35 < TOPE -> Descuenta todo y pasa al siguiente concepto de la siguiente prioridad y realiza de nuevo la pregunta

SI NO

Cantidad a descontar = 35 – TOPE = 2919.68

Es decir solo se pudo descontar la cantidad de 980.32 y el resultado de los movimientos quedaría de la siguiente manera, nótese los estatus de los movimientos de deducciones:

Empleado	Concepto	Tipo movimiento	Monto	Saldos
890078	7	P	3900	X
890078	35	D	980.32	2919.68
890078	36	D	0	1700
890078	39	D	0	6000
890078	195	D	0	1500

Impuesto

En la sección de "Componentes de una formula" se expusieron la base I.S.P.T, GN GE, EX, NI y EXE, incluyendo los @TPG, @EXE y @PGE.

Para el cálculo del impuesto, se tomarán en cuenta los movimientos que sean de estos tipos. Por ejemplo, un empleado que tenga los siguientes movimientos:

Concepto	Descripción del Concepto	Importe	Dato	ISPT	Estatus	@TPG	@PGE	@EXE
PNOR	Percepción Normal	12,120.15	15	GN	A	12,120.15		
AREN	Ayuda De Renta	25.57	0	GN	A	25.57		
DESP	Ayuda De Despensa	328.46	0	EN	A			328.5

Los movimientos PNOR y AREN generan @TPG, esto quiere decir que la base gravable para calcular el impuesto será:

Base Gravable = PNOR (12,120.15) + AREN (25.57)
Base Gravable = 12,145.72

Y DESP es totalmente exento, por lo tanto no se toma para el impuesto, y se acumula como histórico

Base Gravable
Es la cantidad neta con la que se va a calcular el impuesto, y es la suma de todos los movimientos que contengan @TPG

Para el cálculo del impuesto es necesario tener 3 tablas que proporciona la Secretaría de Hacienda y Crédito Público, dichas tablas tienen actualizaciones trimestrales

Tabla del impuesto del articulo 80

Limite Inferior	Limite Superior	Cuota Fija	Porcentaje
0.01	388.76	0	0.03
388.77	3,299.60	11.66	0.1
3,299.61	5,798.76	302.74	0.17
5,798.77	6,740.82	727.61	0.25
6,740.83	8,070.58	963.12	0.32
8,070.59	16,277.22	1,388.64	0.33
16,277.23	47,452.89	4,096.83	0.34
47,452.90	142,358.64	14,696.55	0.35
142,358.65	189,811.54	47,913.58	0.37
189,811.55	9,999,999.00	65,708.41	0.4

Tabla de subsidio

Limite Inferior	Limite Superior	Cuota Fija	Porcentaje
0.01	423.15	0	0.5
423.16	3,591.60	6.34	0.5
3,591.61	6,311.92	164.77	0.5
6,311.93	7,337.33	395.97	0.5
7,337.34	8,784.78	524.17	0.5
8,784.79	17,717.65	755.75	0.4
17,717.66	27,925.45	1,934.91	0.3
27,925.46	35,435.28	2,976.10	0.2
35,435.29	42,522.27	3,486.78	0.1
42,522.28	9,999,999.00	3,727.71	0

Tabla de Crédito al Salario

Limite Inferior	Limite Superior	Cuota Fija	Porcentaje
0.01	1,508.97	347.19	0
1,508.98	2,221.86	347.04	0
2,221.87	2,263.41	347.04	0
2,263.42	2,962.43	346.86	0
2,962.44	3,017.90	335.04	0
3,017.91	3,229.17	326.24	0
3,229.18	3,792.70	326.24	0
3,792.71	4,023.88	302.17	0
4,023.89	4,551.26	277.12	0
4,551.27	5,309.83	251.33	0
5,309.84	6,068.35	216.28	0
6,068.36	6,297.34	185.62	0
6,297.35	9,999,999.00	151.67	0

Cálculo del Impuesto

Una vez que se obtuvo la base gravable, continuaremos con el ejemplo anterior

Base gravable = **12,145.72**

Para cada tipo de Nómina, existe un factor de subsidio:

Nómina	Factor
Agentes de Ventas	0.76
Personal Administrativo	0.76
Personal de Reparto	0.77241

De la tabla de IMPUESTO del articulo 80, se obtiene el límite superior, el inferior, la cuota fija y el porcentaje, suponiendo la base gravable expuesta y que la nómina es del personal administrativo:

Factor = 0.76
Porcentaje = 0.33
Lim inferior = 8070.59
Lim superior = 16,277.22
Cuota fija = 1388.64

Excedente = Base Gravable – Lím inferior = 4075.13
Impuesto marginal A = Excedente * Porcentaje = 1344.7929
Impuesto antes del subsidio = Impuesto marginal A + Cuota fija = 2733.4329

Ahora de la tabla de SUBSIDIO, se obtiene:

Porcentaje = 0.4
Cuota fija = 755.75
Impuesto marginal del subsidio = Impuesto marginal A * Porcentaje = 537.9171
Subsidio = Impuesto marginal del subsidio + Cuota fija = 1293.6671
Subsidio Acreditable = Subsidio – ((1 - Factor) * 2) * Subsidio = 2060.72
IMPTO. A/CRED. AL SAL. = Impuesto antes del subsidio - Subsidio Acreditable = 672.7069

De la tabla CREDITO AL SALARIO se obtiene:
Cuota fija = 151.67
Impuesto Neto = IMPTO. A/CRED. AL SAL - Cuota fija = **1,909.05**

Préstamo de Fondo de Ahorro

Días Trabajados = Si el empleado a trabajado mas de 1 año en la empresa los días trabajados para el fondo de ahorro son 360.
Porcentaje deseado = Al porcentaje que pide el trabajador, este es menor igual a 80%
Porcentaje establecido = 23%
PRES = SDPN * Días trabajados * 0.23 * Porcentaje deseado

Impuesto Anual

El cálculo del impuesto anual, tiene el mismo procedimiento ya explicado, solo que la base gravable se obtiene:

Dato A = (Suma de todos los @TPG + suma de todos los @PGE) del periodo 1 al 22 + suma de todos los @TPG y @PGE de los periodos 23 y 24.

Se calcula el impuesto sobre la base Dato A, y se obtiene un **Impuesto A.**

Impuesto B = suma del impuesto calculado del periodo 1 al 22

Impuesto Total Anual = Impuesto A - Impuesto B

Cálculo de formulas mas elaboradas

Existen algunas formulas que son mas complicadas de calcular, como las que llevan saldos, por lo tanto todos el tiempo dependen de acumulados.

Formulas de Reparto

INFO – INFONAVIT
INFO = SDI * INFO (**incidencia**) * RINF(**días descontados a info**)
RINF = DQUIN(**días trabajados**) – (INCA(**incidencia**) + CAST(**incidencia**) + PESG(**incidencia**))

Formulas de Administrativos

INFO – INFONAVIT
INFO = SDI * INFO (**incidencia**) * RINF(**días descontados a info**)
RINF = DQUIN(**días trabajados**) – (INCA(**incidencia**) + CAST(**incidencia**) + PESG(**incidencia**))

@CSC – Crédito al Salario Calculado
@CSC = Es el dato Crédito al Salario del cálculo del impuesto

@CSP – Crédito al Salario Pagado
@CSP = Resultado @ISP negativo * -1

@ISP – Impuesto Sobre el Producto del Trabajo
@ISP = Cálculo del impuesto (**ya explicado**)

PRES = Préstamo de Fondo de Ahorro
PRES = Cantidad solicitada <= (SDPN * [360 Si DIAST(**antigüedad**) > 360] * 0.25 * (% solicitado no mayor al 0.8))

PFOA = Pago de Fondo de Ahorro
PFOA = (PRES - KPFOA(Acumulado de descuentos de PFOA)) / (QUIN(**número de quincenas a la segunda de noviembre**)
KPFOA = KPFOA(**acumulado**) + PFOA

INTE – Intereses de PFOA
INTE = (PRES – KPFOA) * 0.7475

AFEM - Aportación Fondo de Ahorro Empresa
AFEM = (PNOR +RETR +SOSU) * 0.13

APFO – Aportación Fondo de Ahorro Empleado
APFO = (PNOR +RETR +SOSU) * 0.10

ACFA - Acumulado de Fondo de Ahorro
ACFA = APFO(**acumulado**)+ AFEM(**acumulado**)

SPFOA – Saldo de PFOA
SPFOA = PRES – KPFOA

PALI – Pensión alimenticia
PALI = PALIA(**auxiliar de PALI**) * PALI(**incidencia**)
PALIA = PNOR + DESP + TRAN + @HEX + HRTR + PPVA + BONO + PDOM + @PVA + RETR + COMP - @ISP

PLIQ - Pago de Liquidación
PLIQ = SDPN * 90 + PLIQA(**auxiliar de PLIQ**)
PLIQA = SDPN * 20 * ANTIG(**antigüedad en años**)

Formulas de Agentes

29 - Aportación Fondo de Ahorro Empresa
29 = (888 + 8 + 7 + 12 + 10 + 9 + 815 + 819 + 826) * 0.13
TOPE29 = @SMN * 10 * 30 * 0.13

30 – Aportación Fondo de Ahorro Empleado
30 = (826 + 888 + 8 + 7 + 12 + 10 + 9 + 815 + 819) * 0.10
TOPE29 = @SMN * 10 * 30 * 0.10

31 - Acumulado de Fondo de Ahorro
31 = 29(**acumulado**)+ 30(**acumulado**)

32 – Préstamo de Fondo de Ahorro
32 = Cantidad solicitada <= Total del concepto 31, del año anterior.

Reportes del cálculo de la nómina

Listado de movimientos no aplicados.- Cuando un empleado no laboró determinado número de días, en el transcurso de estos, se debió aplicar los descuento pertinentes, pero como no tuvo percepciones durante ese intervalo de tiempo, estos descuentos no fueron aplicados, sin embargo si fueron calculados y almacenados, se requiere el listado para aplicar adecuadamente estos descuentos.

Impresión de nómina.-Después del cálculo se requiere la impresión de todos los movimientos calculados, agrupando totales de todos los conceptos por:

- Empleado. Clave, Nombre, RFC, num. IMSS, Fecha de ingreso, Salario Diario, total de percepciones, total de deducciones y total a pagar.
- Agencia. Clave, descripción, total de percepciones, total de deducciones y total a pagar.
- División. Clave y descripción.
- Región. Clave y descripción.
- General. total de percepciones, total de deducciones y total a pagar de toda la nómina calculada.

Impresión de recibos.-Existen dos formatos de recibos, uno es para la nómina de Agentes de Ventas, el otro para la nómina del Personal Administrativo y Personal de Reparto, aquí se imprimen todos aquellos movimientos generados por las formulas que tienen la indicación de impresión en recibo, esta indicación se realiza en el momento de crear la fórmula.

Relación de depósitos y cheques.- Obtener los totales a pagar por empleado, especificando quien posee cuenta en el banco y quién no para el pago por cheque.

Layout de depósitos.- Para la nómina del Personal Administrativo, se requieren formatos para los depósitos en Bital(número de cuenta | importe total) e Inbursa:

	nnnnnn
Cve empleado	nnnnnnnnnn
Apellido paterno materno nombre	40 posiciones caracter
Número de cuenta	12 posiciones numéricas
Importe	2 posiciones decimales

Y para la nómina de Agentes de Ventas, el formato para el banco de Santander es:

Región	3 posiciones numéricas de longitud variable
División	3 posiciones numéricas de longitud variable
Agencia	3 posiciones numéricas de longitud variable
Clave del empleado	6 posiciones numéricas de longitud variable
Apellido paterno, materno y nombre	40 posiciones caracter de longitud variable
Importe	n enteros, 2 posiciones decimales

Control de saldos deudores.- Todas aquellas cantidades que hacen falta descontarse, y las cuales se tomarán en cuenta para las nóminas posteriores, hasta que se liquiden

Cálculo de sábanas de percepciones para actualizar salarios integrados.-

Agentes de ventas

Para el cálculo del Salario Integrado es necesario calcular antes el siguiente factor:
Ayuda de despensa = Suma de movimientos del concepto 59
Fondo de Ahorro Empresa = Suma de movimientos del concepto 29
Conceptos IMSS, SAR, INFONAVIT = Suma de movimientos diferentes a 29 y 59
Factor = (Salario Mínimo * 0.4) * 30.4
Si la ayuda de despensa es mayor al factor calculado, entonces, se calcula el excedente
Excedente = Ayuda de despensa - Factor
El 3% de la Aportación Empresa = (Fondo de Ahorro Empresa / 0.13) * 0.03

Salario Integrado Agentes de Ventas = (El 3% de la Aportación Empresa + Excedente + Conceptos IMSS, SAR, INFONAVIT)

Impresión de cheques.- Formato requerido según los cheques de ADSA S.A de C.V.

Listado de movimientos.-Puede realizarse un reporte de cualquier concepto existente de cualquier periodo y de cualquier año, desglosándolo así por totales de movimiento.

Bitácora de cálculo.- Es el seguimiento del cálculo realizado, desglosando el resultado de cada operando de la formula que generó el movimiento a consultar, si este es el impuesto, se desglosan los intervalos en los que cayó la base gravable, en las tablas del impuesto del artículo 80 explicado anteriormente, así como los factores e importes intermedios obtenidos durante este proceso.

Mantenimiento a los movimientos.- Una vez que se realizó el proceso del cálculo de la nómina, se procede a la revisión, consultando los movimientos calculados por empleado, dónde es posible corregir cálculos finales, esto es bajo la responsabilidad del usuario, se guarda un histórico de cambios realizados, por este usuario, para cualquier aclaración.

Actualización de acumulados.- Informe en pantalla de todo el histórico de un concepto durante el año, desglosando las cantidades por mes.

Acumular periodo.- Una vez que ya se autorizó la nómina, se acumularán todos los movimientos generados en ese periodo, para que tengan efecto en aquellas formulas que los utilizan.

Nóminas Especiales.- Estas nóminas se realizan en caso de:

- Nóminas complementarias
- Pago reparto de utilidades
- Casos especiales como el pago de conceptos fuera de las fechas establecidas de pago.
-

Cuándo se realiza una nómina especial, se construye a partir de las nóminas existentes, es decir, sus componentes como son; fórmulas y empleados, se encuentran obtenidas originalmente en las nóminas normales. El proceso de cálculo es el mismo así como también el acumulado de sus periodos.

2.7.4.3 Procesos Especiales

Reparto de utilidades (Proceso PTU)

Es el cálculo de reparto de utilidades, para esto es necesario de unos factores, los cuales se actualizan cada vez que se realiza este proceso.

Los empleados que tienen derecho al pago de utilidades son todos aquellos que tienen la planta, y en caso del personal eventual, ellos tendrán que haber trabajado mínimo 60 días continuos o no, en el transcurso del año, es decir, si un empleado ha sido contratado varias veces durante el año, y los intervalos de tiempo trabajado suman 60 días, el empleado tiene derecho al pago de PTU.

El cálculo se realiza en base a los factores y a los conceptos deseados:

Año	Factor Importe	Factor Dato
1999	0.00110505	0.37019691
2001	0.7	0.8

Base = Dato de contabilidad.
Dato A = Número de días trabajados de todos los empleados de las tres nóminas.
Dato B = Sumatoria de todas las percepciones de las tres nóminas, especificando cuáles se tomarán en cuenta.

Factor A = (Base / 2) / Dato A
Factor B = (Base / 2) / Dato B

Importe A = Factor A * Número de días trabajados por empleado durante el año.
Importe B = Factor B * Sumatoria de percepciones por empleado durante al año.

PTU a pagar:

Importe Total = Importe A + Importe B

Para el cálculo del impuesto:

El procedimiento para calcular el impuesto en PTU, es el mismo para aguinaldo, en base al Art. 86.

Al importe total se le restan 15 días de salario mínimo exentos de impuesto:

Base gravable PTU = Importe Total – (40.35 * 15) para el Art. 86

Según la fracción I del artículo 86, se obtiene la cantidad mensual:

Base ISPT = (Base gravable PTU / 365) * 30.4

Según la fracción II del artículo 86, se obtiene la base gravable total:

Base Total = Base ISPT + Ingreso Mensual del trabajador

Se procede a calcular el impuesto (explicado en el apartado cálculo del impuesto) con la Base Total obtenida, como resultado tenemos un **Impuesto A.**

Se procede a calcular el impuesto con la Base ISPT normal mensual del empleado, como resultado tenemos un **Impuesto B**. En caso de la nómina de Agentes de ventas, para obtener la Base ISPT normal mensual:

Promedio de comisiones

Base Gravable = Suma de Comisiones de Mayo del año anterior a Abril del año en curso / 12

Según la fracción III del artículo 86, se obtiene la diferencia de los impuestos calculados:

Diferencia = Impuesto A - Impuesto B

Según la fracción V se obtiene el cociente de la Diferencia y la Base ISPT de la fracción I

Factor = Diferencia / Base ISPT

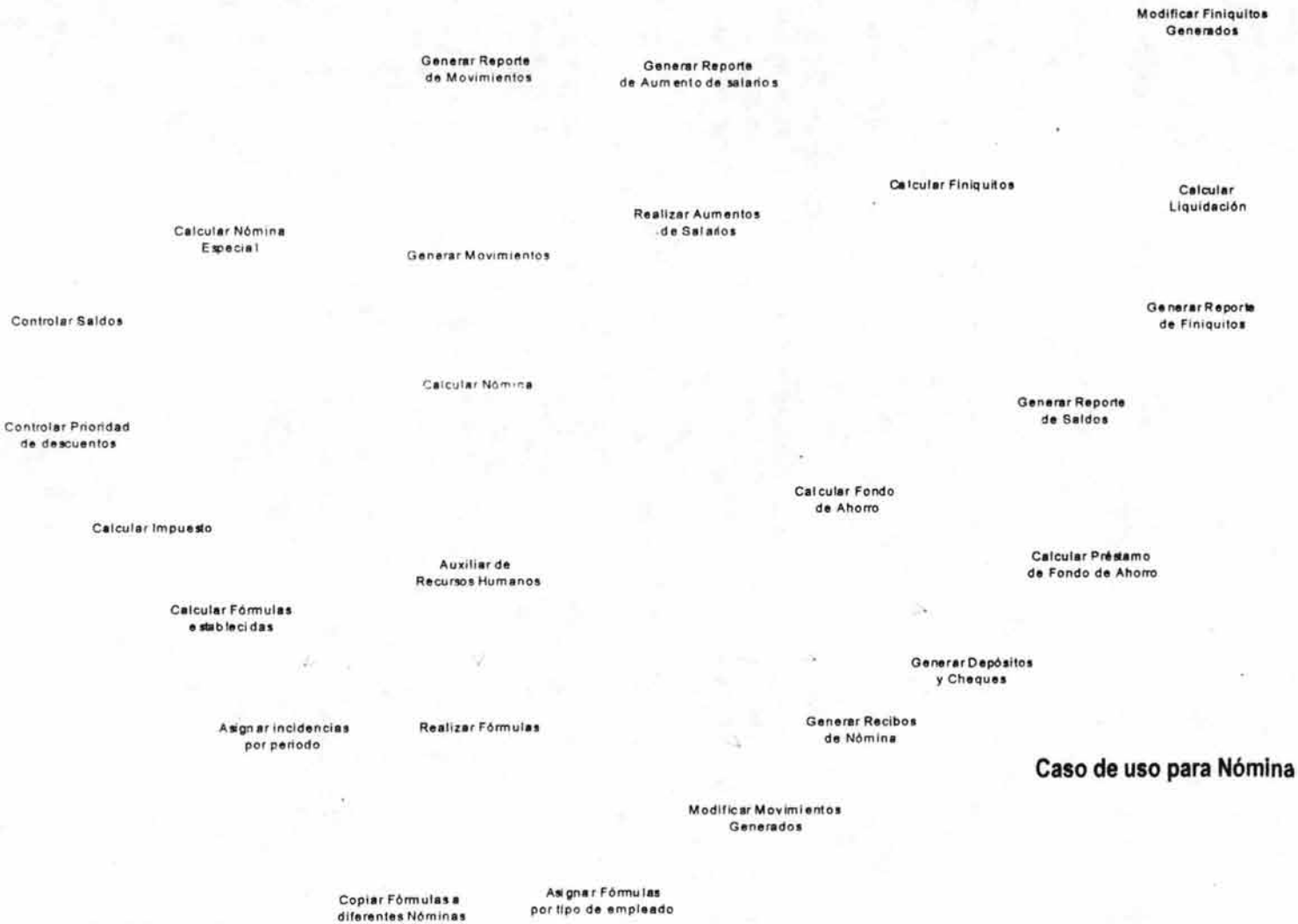
Según la fracción IV se obtiene el impuesto total, siendo este el producto del cociente de la fracción V y LA Base gravable PTU

Impuesto Total = Factor * Base gravable PTU

Aumento de salarios

Existe un aumento de salarios cada año, después de la revisión contractual que realiza la empresa con el sindicato, o bien aumento de sueldos por diferentes causas, este se puede aplicar por:

- Porcentaje
- Cantidad
- De forma individual o general



Caso de uso para Nómina

2.7.4.4 IMSS

Impresión de listados

Es la impresión de los movimientos tomados en cuenta por el IMSS:

- Altas
- Bajas
- Modificaciones de salario
- Faltas

Histórico de sueldos

Es el almacenamiento de todos los cambios de salario del empleado, desde su fecha de ingreso, incluyendo así:

- Salario diario
- Salario IMSS
- Salario INFONAVIT
- Salario SAR
- Salario Real
- Cambios de puesto
- Cambios de tipos de empleado

Listado de transferencia ASCII a SUA.

Se obtiene un listado de movimientos ya mencionado, dónde se encuentran las altas, bajas, modificaciones de salario y faltas, pasa por una revisión, para así aplicarle ajustes de acuerdo a las cantidades de la nómina calculada, posteriormente se genera un archivo con el formato especificado por el IMSS, para el programa SUA que es el que genera todos los pagos correspondientes.

Formato requerido por el IMSS:

CAMPO	TIPO	LONG.	TIPO DE MOVIMIENTO				
			02	07	08 y 09	11	12
Registro Patronal IMSS	X	11	x	x	x	x	x
Num. de Seguridad Social	9	11	x	x	x	x	x
Tipo de Movimiento	9	2	x	x	x	x	x
Fecha de Movto. (ddmmaaa)	9	8	x	x	x	x	x
Folio Incapacidad	X	8					x
(XX999999)	9	2				x	x
Días de la Incidencia							
Salario Diario Integrado o Aportación Voluntaria	9	5,2		x	x		

Criterios:
El registro debe tener una longitud fija de 49 caracteres.

Tipos de movimientos:

02 Baja
07 Modificación de Salario
08 Reingreso
09 Aportación Voluntaria
11 Ausentismo
12 Incapacidad
X =Campos Alfanuméricos
9 =Campos Numéricos

El Salario Diario Integrado o Aportación Voluntaria (5 enteros y 2 decimales) debe grabarse SIN punto decimal.
Utilice sólo MAYUSCULAS para los caracteres alfabéticos y en caso de que su equipo no cuente con el caracter Ñ use /.
Los campos numéricos se deben complementar con ceros a la izquierda.

Histórico de movimientos IMSS

Es el almacenamiento de movimientos del IMSS antes mencionados, por empleado

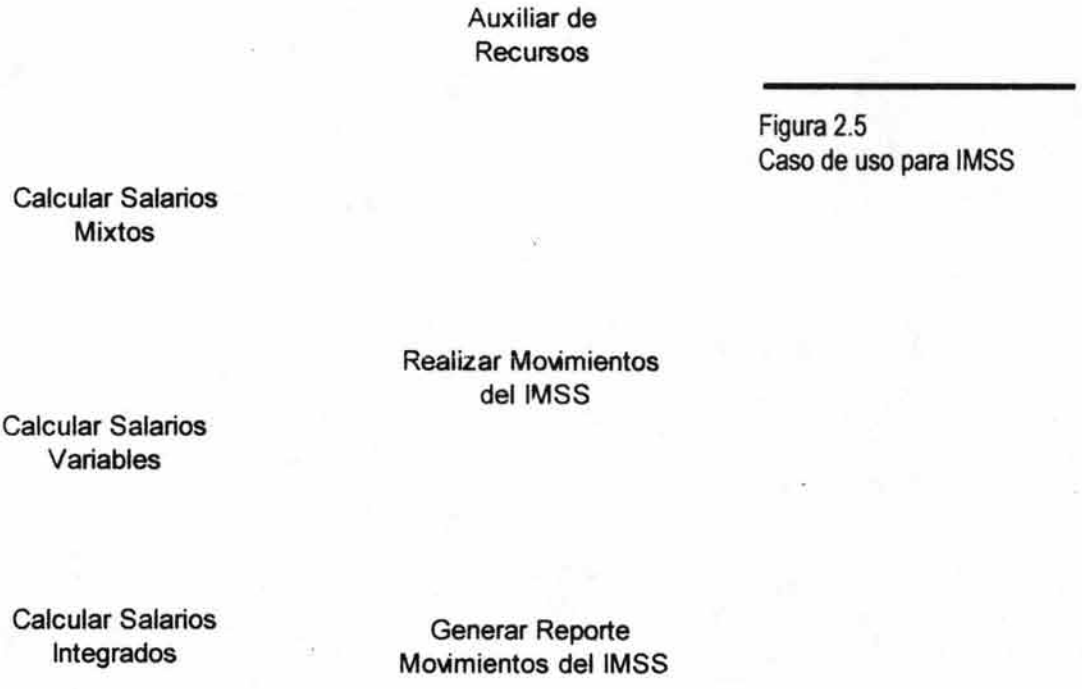


Figura 2.5
Caso de uso para IMSS

2.7.4.5 Fondo de Ahorro

El Fondo de ahorro es una cantidad que está compuesta por una aportación de la empresa mas una aportación del empleado, este se va acumulando y se entrega en Diciembre, el cálculo ya se expuso en el cálculo de la fórmula

Cálculo de préstamos

Se calcula lo más que puede pedir un empleado:

%R = Porcentaje requerido (hasta 80%)
DIAS = Días trabajados, si el empleado ha trabajado mas de 360, los días se topan en esa cantidad.
F = Factor de Fondo de Ahorro = 0.23
Total de Préstamo = SDPN * DIAS * F * %R

Esto es para todas las nóminas.

2.7.4.6 Finiquitos y Liquidaciones

Para la nómina del Personal Administrativo

Cálculo del finiquito

Para el cálculo del finiquito es necesario los siguientes datos, recordando que las prestaciones a pagar son de cada año, por lo tanto, se obtendrán los acumulados de los conceptos del año en curso.

Nombre del empleado
Clave del empleado
Fecha de Alta
Antigüedad = (Fecha de Baja – Fecha de alta) / 360
Fecha de Baja
SDPN = Sueldo Diario Percepción Normal
Puesto
División

Percepciones:

Posibles percepciones a agregar como BONO, COMP, etc.
Suma de ACFA
Suma de @GRA
Suma de COME
Suma de PVAC
Suma de @VAC(vacaciones no disfrutadas)

Deducciones:

PFOA Total
SEAU
ISPT Total

Dónde el ISPT Total se calcula:
SMI = Salario mínimo.

Impuesto A con Art. 80

Base Gravable = [@GRA – (SMI * 30)] + [PVA – (SMI * 15)]
Dónde [PVA – (SMI * 15)] se aplicará siempre y cuándo no se haya aplicado durante el año.

Impuesto B CON Art. 80
Base Gravable = COME + PVAC + (SDPN * 30)

Impuesto Bi Art. 80
Base Gravable Bi = (SDPN * 30)
Impuesto B = Impuesto B + Impuesto Bi

Si existe PNOR en la quincena actual entonces se calcula un impuesto D

Impuesto C = (**Impuesto Bi** / 30) * Dato de PNOR

ISPT Total = Impuesto A + Impuesto B + Impuesto C

Cálculo de liquidación.- Es el mismo procedimiento que para finiquitos, solo que en percepciones se agregan:

PANT(Prima de Antigüedad) = (Antigüedad * 12) * SDPN
PLIQ(Pago de Liquidación) = (Antigüedad * 20* SDPN

Y para calcular ISPT Total se agrega el impuesto:

Impuesto D
Base Gravable = (PLIQ + PANT) – [(SDPN * 90) * Antigüedad en años enteros]
Dónde la Antigüedad en años enteros se refiere al siguiente criterio:

Si Antigüedad = 0.5 Antigüedad = 0
Si Antigüedad = 0.6 Antigüedad = 1

Factor = **Impuesto Bi** / (SDPN * 30)
Impuesto D = Base Gravable * Factor

ISPT Total = Impuesto A + Impuesto B + Impuesto C + Impuesto D

Cálculo de liquidación con SDI.- En el cálculo de PANT y PLIQ se toma en cuenta el SDI y no el SDPN, el resto es el mismo procedimiento mencionado.

Mantenimiento de finiquitos.- Una vez calculado el finiquito, talvez se desea realizar una modificación manual en algunas cantidades, esto repercute en el cálculo, es por eso que en este apartado se podrá recalcular el finiquito después de las modificaciones o bien dejarlo sin recalcular.

Actualización de Acumulados.-Una vez que fue aceptado el cálculo del finiquito, se procede a acumularlo junto con los demás movimientos de nómina.

2.7.4.7 Reportes

Archivo SUA. Explicado en el aparatado IMSS.

HSBC Fondo de Ahorro

Btal.txt

Número de cuenta	Numérico
,	Separador (,)
Monto total a pagar	Numérico decimal ###,###.##

Inbursa Fondo de Ahorro

Inbursa.txt

Consecutivo	Numérico
Apellido paterno	Cadena
Apellido materno	Cadena
Nombre	Cadena
Número de cuenta	Numérico
Monto total a pagar	Numérico decimal ###,###.##
,	Separador (,)

General Fondo de Ahorro

GeneralFA.xls

Número de Empleado
Nombre Completo del empleado
Número de periodo
Aportación Empleado
Aportación empresa
Monto de préstamo
Descuentos
Intereses

Altas

Altas.xls

Consecutivo
Número de Afiliación
Número de División
CURP
Nombre completo del empleado
Salario Mensual
Fecha de Alta
Número de empleado

Bajas

Bajas.xls

Consecutivo
Número de Afiliación
Número de División
CURP
Nombre completo del empleado
Salario Mensual
Fecha de Baja
Número de empleado

Faltas

Faltas.xls

Consecutivo
Número de Afiliación
Número de División
CURP
Nombre completo del empleado
Salario Mensual
Fecha de Falta
Número de empleado

Cambios de salario

CambioS.xls

Consecutivo
Número de Afiliación
Número de División
CURP
Nombre completo del empleado
Nuevo Salario Mensual
Fecha de Cambio de Salario
Número de empleado

Depósitos Bital

Bital.xls

Número de cuenta
Monto total

Depósitos Inbursa

Inbursa.xls

Consecutivo	Numérico	5 posiciones
Número de empleado	Numérico	10 posiciones
Nombre completo del empleado	Cadena	
Número de cuenta	Numérico	12 posiciones
Importe neto	Numérico	###,###.##

Depósitos nómina de confianza administrativos

DepositosConfianza.xls

Nombre de la Agencia
Número de empleado
Nombre completo del empleado
Número de cuenta
Importe neto

Depósitos nómina de sindicalizados administrativos

DepositosSindicalizados.xls

Nombre de la Agencia
Número de empleado
Nombre completo del empleado
Número de cuenta
Importe neto

Depósitos General

DepositosGeneral.xls

Nombre del banco
Nombre de la Agencia
Número de empleado
Nombre completo del empleado
Número de cuenta
Importe neto

Alta IMSS

Según Formato IMSS

Baja IMSS

Según Formato IMSS

Incapacidades

Incapacidades.xls

Consecutivo
Número de Afiliación
Número de División
CURP
Nombre completo del empleado
Salario Mensual
Fecha de Incapacidad
Número de empleado

Salarios IMSS

SalariosIMSS.xls

Número de empleado
Nombre completo del em`pleado
Nombre del tipo de salario
Salario Integrado anterior
Salario Diario
Factor IMSS
Ayuda de transporte
Salario Variable
Nuevo salario Integrado

2.7.5 Análisis del Módulo de Inventario de Recursos Humanos

El inventario de RH no es mas que la existencia de todos aquellos datos relacionados con el empleado. Desde sus datos personales hasta los datos que tiene o tendrá con la empresa, en este apartado se expondrán todos estos datos en un organigrama proporcionado por la misma empresa.

Figura 2.6
Inventario de Recursos Humanos

Datos Generales =
Datos Personales +
Escolaridad / Capacitación +
Trayectoria Laboral +
Datos de apoyo +
Datos económicos +

Datos Personales =
Nombre del Empleado +
Dirección del Empleado +
Teléfono propio +
Teléfono recados +
Fecha de nacimiento +
Lugar de Nacimiento +
Edad +
Sexo +
Estado Civil +
Nacionalidad +
Cartilla +
CURP +
RFC +
IMSS +
Zona económica (1,2,3)

Nombre del Empleado =
Nombre +
Apellido Paterno +
Apellido Materno +

Dirección del Empleado =
Calle +
Número Exterior +
Número Interior +
Colonia +
Delegación +
Población +
Código Postal

Escolaridad / Capacitación =
Datos Escolares +
Conocimientos generales +
Capacitación

Datos Escolares =
Escuela +
Años +
Terminó +
Nivel (Primaria, Secundaria, Bachillerato, Licenciatura, Posgrado) +
Horario

Conocimientos Generales =
Idioma +
Conocimientos Secretariales +
Conocimientos Computacionales +

Capacitación =
Curso +
Duración +

Impartido en +
Promovido por +
Áreas de interés de desarrollo

Trayectoria Laboral = Empleo

Empleo = Nombre de la empresa +
Fecha de ingreso +
Fecha de egreso +
Domicilio +
Teléfono +
Puestos desempeñados +
Sueldo Inicial +
Sueldo Final +
Nombre del Jefe directo +
Motivo de separación +

Datos de Apoyo = Familiar +
Referencia personal

Familiar = Nombre del padre +
Nombre de la madre +
Nombre de los hermanos +
Nombre del (la) esposa +
Nombre de los hijos
Nombre de los dependientes económicos
Parentesco +
Edad +
Sexo +
Medio de enterado por empleo +
Familiar en la empresa +
Puede viajar +
Puede cambiar de residencia +
Esta afiliado a un sindicato +
Deportes que practica +
Referencia personal +
Avisar a en caso de accidentes

Referencia personal = Nombre +
Domicilio +
Ocupación +
Teléfono

Datos Económicos = Tienen Crédito INFONAVIT +
Crédito Hipotecario +
Monto mensual Otro Ingreso +
Ingreso Mensual Conyuge +
Valor aproximado de propiedades +
Automóvil +
Deudas +
Gasto Mensual +
Tiene Casa Propia +

Automóvil =	Marca +
	Modelo
Deudas =	A quién debe +
	Importe Total +
	Abono Mensual
Datos en la empresa =	Estructura Organizacional +
	Sueldos +
	IMSS / INFONAVIT +
	Ingresos / Reingresos +
	Cambios de puesto
Estructura Organizacional =	Fecha de Ingreso +
	Fecha de planta +
	Puesto +
	Centro de costo +
	Tipo de nómina (Confianza, Sindicalizado)+
	Tipo de Empleado +
	Tipo de contrato +
	Nivel +
	Horario +
	Reporta a +
Sueldos =	Salario Diario +
	Salario Mensual +
	Salario Diario Integrado IMSS +
	Salario Diario Integrado SAR +
	Salario Diario Integrado INFONAVIT +
	Salario Diario Integrado REAL +
	Número de cuenta +
	Tipo de pago +
	Histórico de sueldos
IMSS / INFONAVIT =	Número de Afiliación +
	Registro patronal +
	Tipo de salario (Fijo, Mixto, variable)+
	Número de Crédito de INFONAVIT +
	Porcentaje +
	Cantidad Fija
Ingresos / Reingresos	Fecha de alta +
	Baja de puesto +
	Centro de costo +
Cambios de Puestos	Fecha de Cambio +
	Centro de Costo +
	Motivo del cambio
Centro de costo =	Regón +
	División +
	Agencia

Como podemos Observar se encuentran todos los datos que se requieren del empleado, en algunos apartados existe redundancia en otros se encuentra mal expuesto, para el diseño de los datos que se verán en el capítulo siguiente, nos ayudaremos del capítulo II dónde se encuentra la demás información.

2.7.6 Análisis del Módulo de Catálogos Generales

Información Acerca del empleado

Nivel.	Contiene el nivel del empleado en la estructura organizacional
Puesto.	Contiene los puestos existentes en la empresa.
Tipo de pago.	Contiene los diferentes tipos de pago, ya que este puede ser cheque, efectivo, depósito, etc.
Tipo de empleado.	Contiene si el empleado es de confianza, sindicalizado o es aspirante u otro.
Estado Civil.	Contiene el estado civil que puede tener el empleado, Soltero, viudo, divorciado(a), etc.
Causa de baja.	Contiene las posibles causas por las que se dio de baja el empleado
Tipo de Registro.	Contiene los tipos de registros que puede tener un empleado, como su RFC, CURP, etc.
Tipo de Contrato.	El contrato puede ser Nuevo o viejo, de acuerdo a la diferencia de prestaciones, antes y después del 1 de Junio de 1991.
Tiempo de contrato.	Contiene si el contrato es eventual, de planta, temporal por alguna causa, etc.
Tipo de Salario.	Contiene los diferentes tipos de salario como Real, Integrado, IMSS, etc.
Región.	Contiene todas las regiones existentes.
División.	Contiene todas las divisiones existentes.
Agencia.	Contiene todas las agencias existentes.
Banco.	Contiene las claves y el nombre de los bancos en los cuales se realizan los depósitos de nómina.
Zona económica.	Contiene todas las zonas económicas existentes.
Tipo de Examen.	Contiene todos los tipos de exámenes que se aplican a los aspirantes, como psicométrico, técnico, etc.

Información sobre la Nómina

Compañía.	Contiene todas las compañías que gestiona el Sistema.
Tipo de Nómina.	Contiene si la nómina es especial o normal.
Vacaciones.	Contiene la tabla de vacaciones según la antigüedad y la nómina.
Periodo.	Contiene los periodos de cada nómina.
Concepto.	Contiene todos los conceptos existentes por nómina.
Tipo de Concepto.	Contiene si el concepto es deducción, percepción, etc.
Aguinaldo.	Contiene el aguinaldo por antigüedad, por nómina.
Tabla Impuesto.	Contiene las tablas para poder calcular los impuestos.

Factor.	Contiene los diferentes factores, como factores al salario integrado, factor al subsidio, etc.

Información sobre Formula

Operando fuente.	Contiene todas aquellas variables que se utilizan en el cálculo de formulas.
Base I.S.P.T.	Contiene las claves de la Base I.S.P.T, como son @PGE, @EXE, etc.
Tipo de Tope.	Al topar el total de un concepto, este puede ser Mensual, Quincenal, etc.

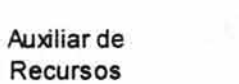
Administración del Sistema

Grupo.	Contiene todos los grupos con permisos existentes.
Sistema.	Contiene el nombre de todos los sistemas existentes.
Objeto Interfaz.	Este catálogo es para uso exclusivo de los desarrolladores, contiene las claves de todos los controles de las ventanas.

Capacitación

Equipo material.	Contiene todo el equipo que posee el departamento de Capacitación.
Aula.	Contiene todas las aulas existentes.
Tipo de Curso.	Contiene si el curso es externo interno, etc.
Tipo de evaluación.	Contiene si la evaluación del alumno es parcial, total, etc.

Figura 2.6
Caso de uso para IMSS



2.8 Administración del Sistema

2.8.1 Seguridad del sistema

En este apartado se requiere que el usuario con los permisos correspondientes, pueda asignar, borrar y modificar los permisos de todos aquellos usuarios que tengan acceso al sistema.

Existen grupos con permisos ya establecidos, es decir, grupos de personas que tienen igual número de permisos, y estos son iguales, de tal manera, que cuando se de alta un nuevo usuario y se le asigne a un grupo, el usuario tendrá todos los permisos que tiene ese grupo.

Se requiere saber del empleado:

Empleado = Código de empleado +
Nombre de usuario +
Contraseña +

Permisos = Apartados del sistema dónde puede acceder el empleado
Grupo = Conjunto de permisos

Veamos nuestro análisis en una caso de uso en la Fig 2.7.

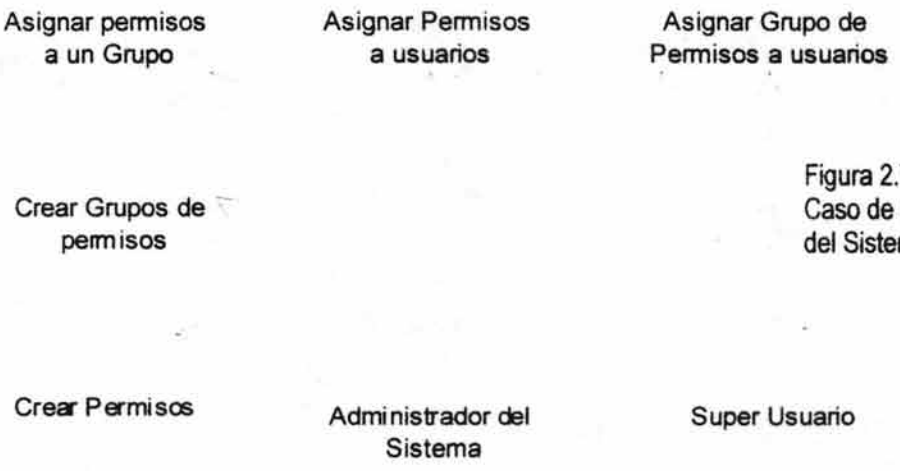


Figura 2.7
Caso de uso para Seguridad del Sistema

2.8.2 Compañías

En este apartado se crearán las compañías necesarias que utilizará el sistema, este mantenimiento solo lo podrán realizar los súper usuarios[38] correspondientes. La información que se almnacenará de cada compañía es la siguiente:

Compañía = Código de la compañía +
Nombre Comercial +
Razón Social +
RFC +
Registro ante el IMSS +
Dirección Fiscal

Dirección Fiscal = Código de la compañía +
País +
Estado +
Delegación o Municipio +
Código Postal +
Colonia +
Calle +
Número exterior +
Número exterior

A estos datos se encontrarán enlazadas las nóminas de cada compañía.

[38] Son aquellos usuarios que tienen privilegios especiales en el sistema, como los gerentes o directivos, estos también cuentan con diferentes niveles.

Capítulo III
Diseño del SRH

3.1. Diseño de la Base de Datos

En el diseño del sistema veremos como se va a estructurar todo el flujo de información, comenzando por el Diseño de la Base de Datos, por medio de métodos ya establecidos, así como el modelado de objetos que gestionarán partes generales y específicas del Sistema, en esta etapa podemos decir que se realizará el sistema en sí, ya que habiendo pasado por la etapa de análisis, el hacer un buen diseño del sistema significa que se facilitará en gran medida el desarrollo del mismo, incluso nos puede llevar mas tiempo que el que nos llevamos en el análisis y diseño, en muchos casos.

3.1.1. Modelo Conceptual de Datos (MCD)

En esta parte podemos tener una visión global del universo existente de datos de nuestra empresa, lo mas conveniente es esquematizarlo. El departamento de Recursos Humanos, como se mencionó en el capítulo I, se basa en la administración de todo el personal de la empresa, esto se traduce en el almacenamiento de toda la información sobre todos y cada uno de los empleados, como también de los que todavía no lo son, por lo tanto la base de nuestro modelo de datos son los empleados y los aspirantes.

Entonces comenzaremos por realizar el Modelo conceptual de datos, en base al análisis realizado en el Capítulo II, podemos visualizar en un plano general lo que va hacer nuestro modelo conceptual. En la Figura 3.1 observamos este modelo realizado en Microsoft Visio 2000, dónde planteamos los datos generales para el diseño del sistema.

Figura 3.1
Modelo Conceptual Global

Ahora cada entidad se desglosará en sus entidades correspondientes, dónde cada entidad contiene sus propios atributos, los cuales se encuentran en el Archivo "Modelo Conceptual.VSD".

Al modelo conceptual decidí dividirlo en 5 módulos:

- EMPLEADO. Se encuentran todos aquellos datos concernientes al empleado, es decir, datos personales como los datos que tiene o tendrá dentro de la empresa, aquí también se encuentra la gestión que tendrá el apartado Reclutamiento y Selección.
- NÓMINA
- FORMULA
- ADMINISTRACIÓN DEL SISTEMA
- CAPACITACION

La entidad empleado se relaciona con la gran mayoría de las demás entidades, ya que siendo esta la entidad mas importante y es de aquí de dónde parten todos los demás procesos y mantenimiento de nuestros datos.

El siguiente modelo conceptual nos dará la pauta para nuestro desarrollo de sistema, este es muy importante ya que como se mencionó en el capítulo I (Modelo Entidad - Relación) será nuestra base para el diseño de nuestra base de datos. Este modelo se basa en el capítulo II Análisis de los Requerimientos, continuando así con las etapas del desarrollo de un sistema.

EMPLEADO I

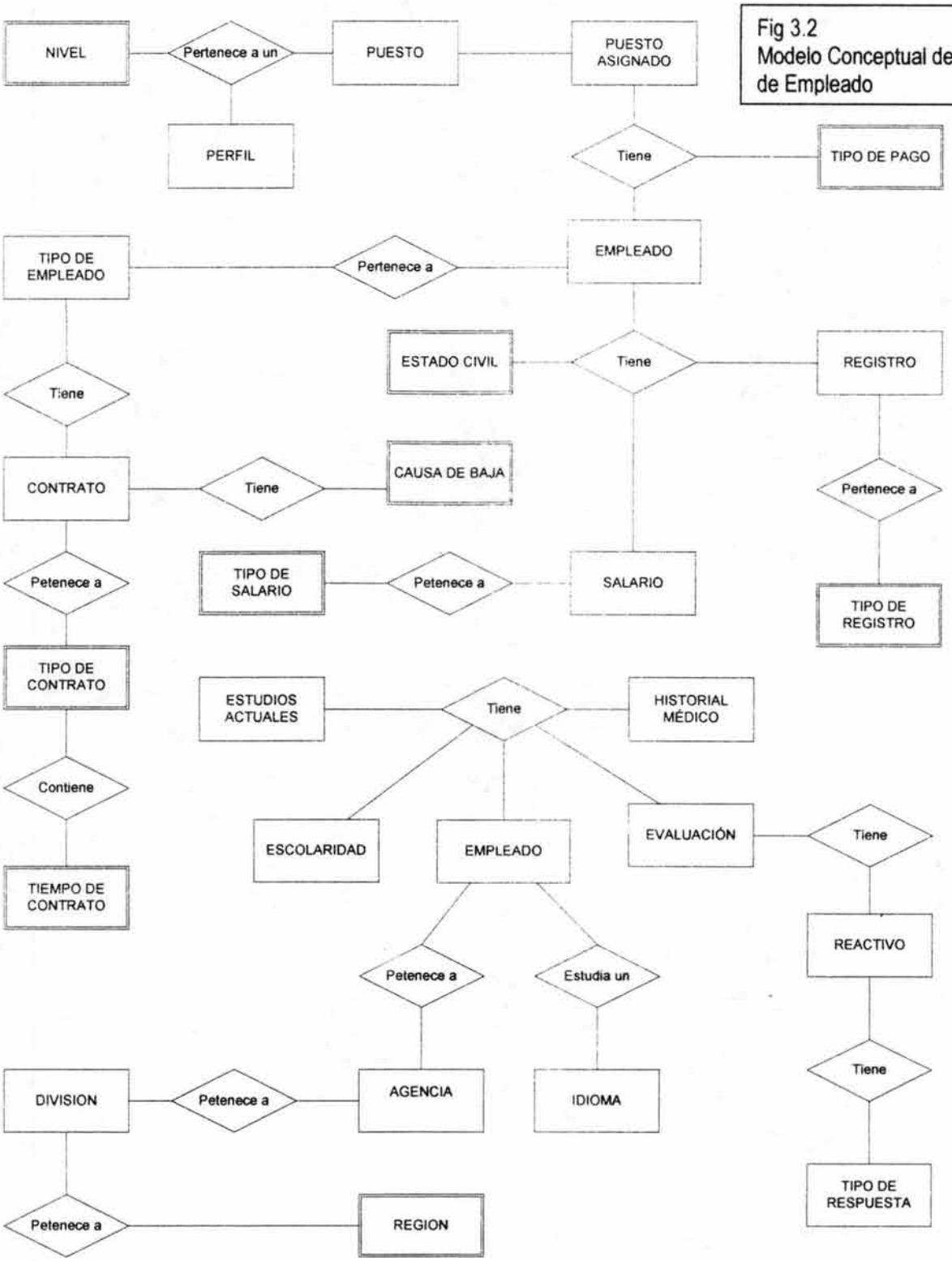


Fig 3.2
Modelo Conceptual de Datos de Empleado

Módulo EMPLEADO II

Fig 3.3
Modelo Conceptual de Datos de Empleado

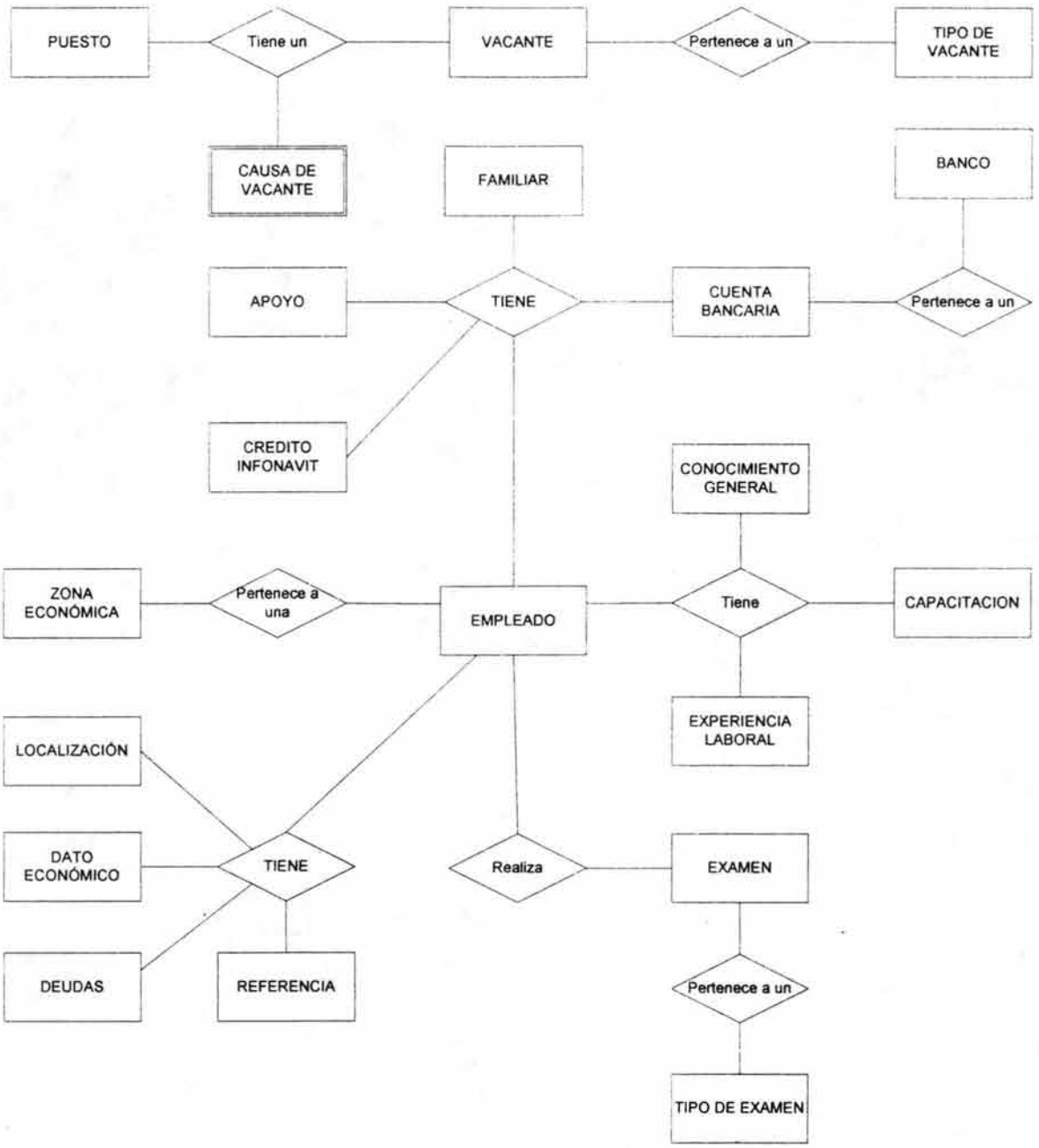
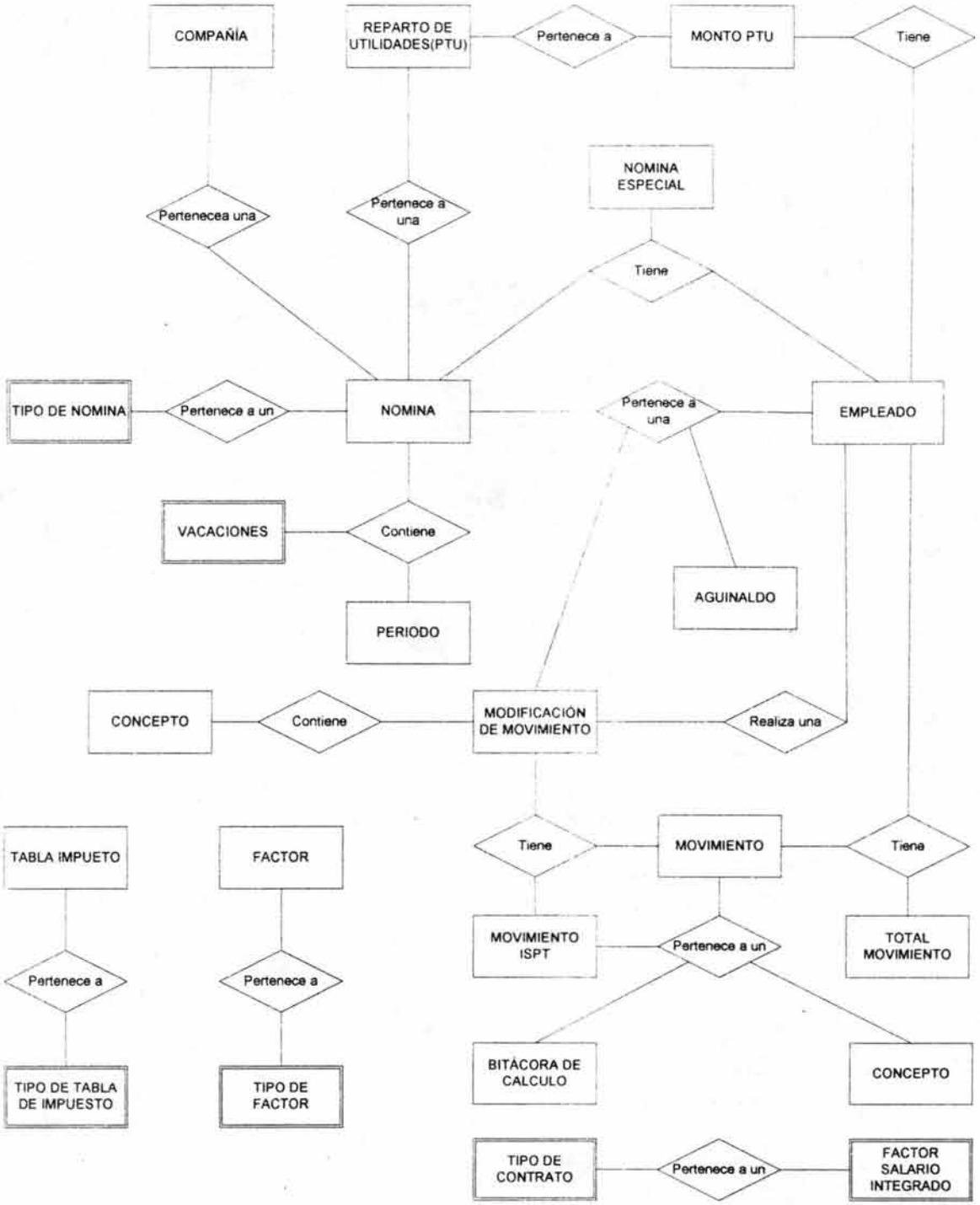


Fig 3.4
Modelo Conceptual de Datos de Nomina

NÓMINA

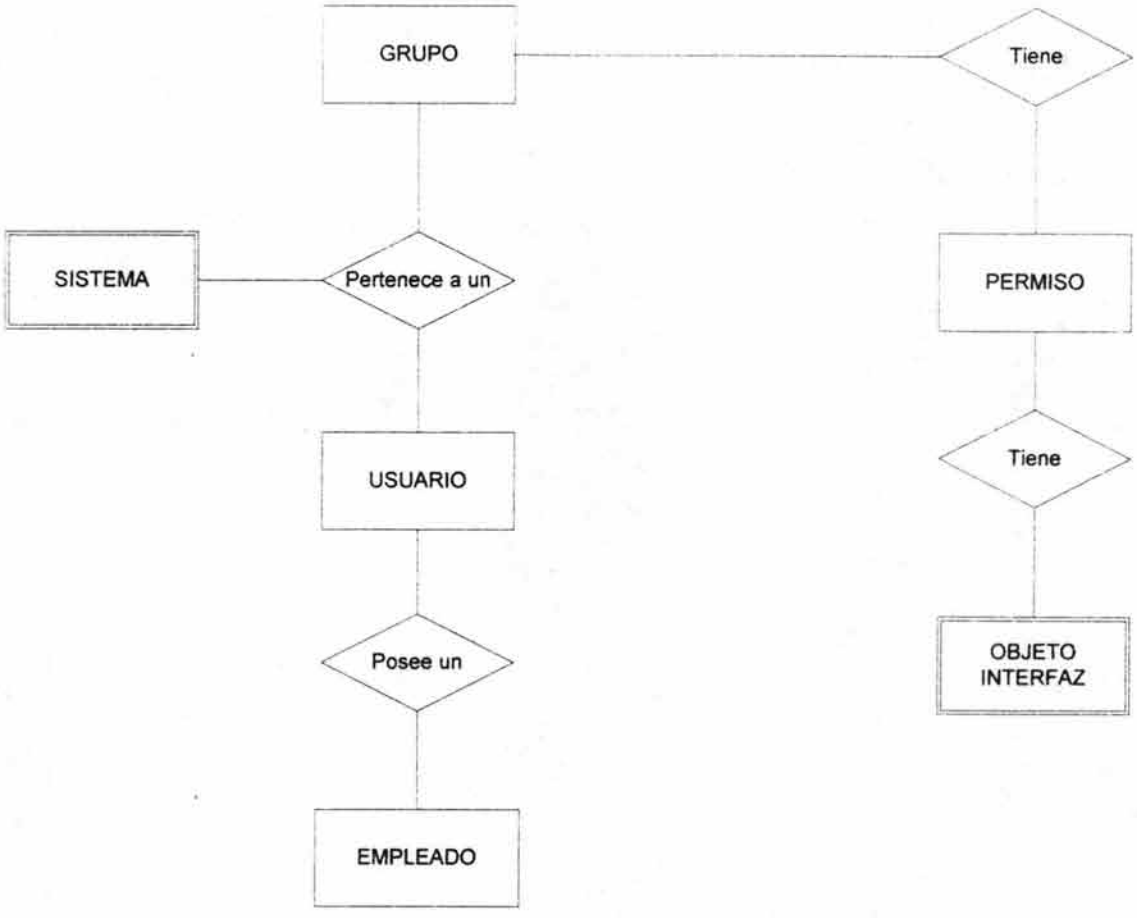

COMPAÑÍA
REPARTO DE UTILIDADES(PTU)
Pertenece a
MONTO PTU
Tiene
NOMINA ESPECIAL
Pertenecea una
Pertenece a una
Tiene
TIPO DE NOMINA
Pertenece a un
NOMINA
Pertenece a una
EMPLEADO
VACACIONES
Contiene
AGUINALDO
PERIODO
CONCEPTO
Contiene
MODIFICACIÓN DE MOVIMIENTO
Realiza una
TABLA IMPUETO
FACTOR
Tiene
MOVIMIENTO
Tiene
Pertenece a
Pertenece a
MOVIMIENTO ISPT
Pertenece a un
TOTAL MOVIMIENTO
TIPO DE TABLA DE IMPUESTO
TIPO DE FACTOR
BITÁCORA DE CALCULO
CONCEPTO
TIPO DE CONTRATO
Pertenece a un
FACTOR SALARIO INTEGRADO

FORMULA

Fig 3.5
Modelo Conceptual de Datos
de Fórmula

Fig 3.6
Modelo Conceptual de Datos
de Administración de Sistema

ADMINISTRACIÓN DEL SISTEMA



CAPACITACIÓN

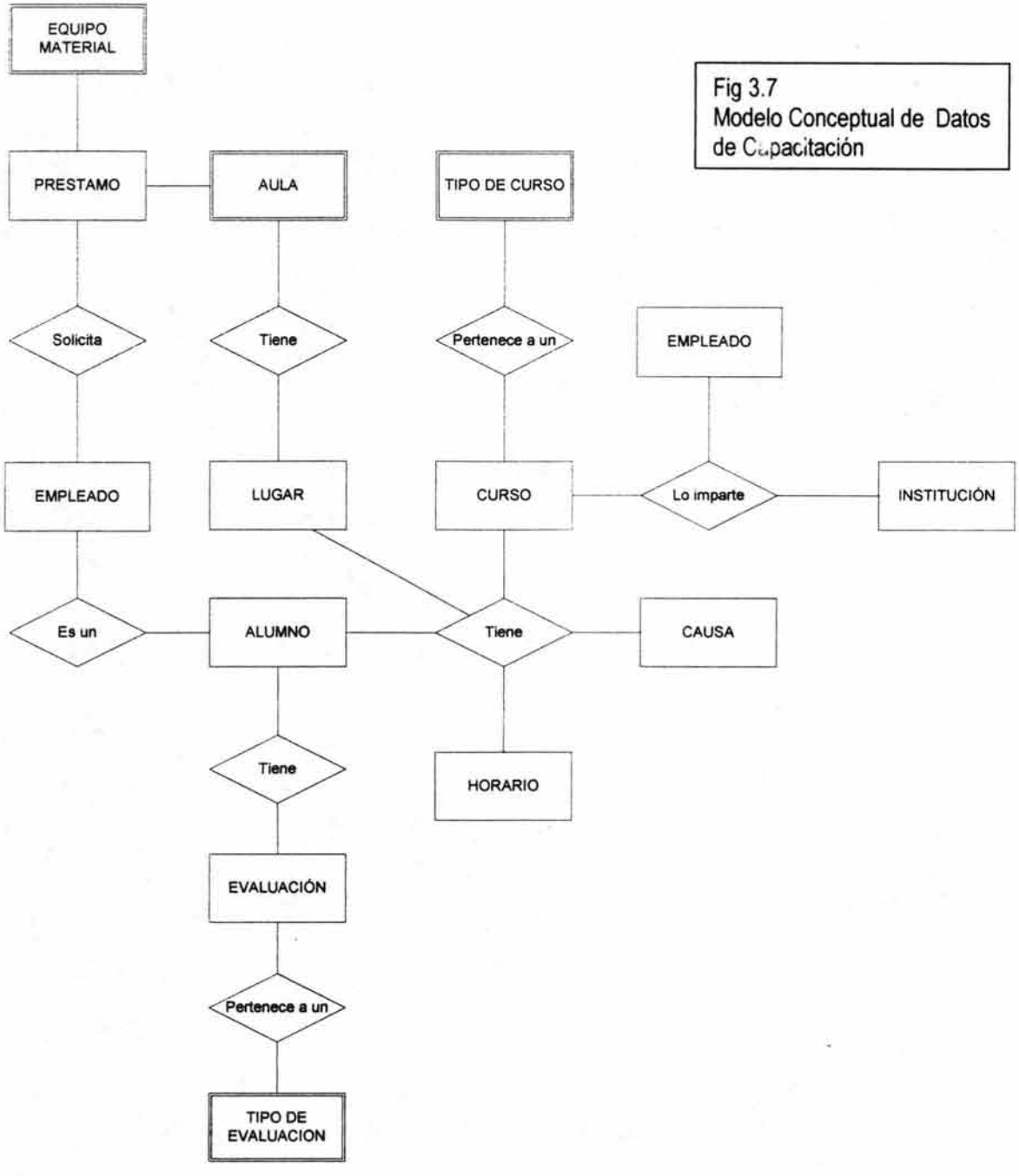


Fig 3.7
Modelo Conceptual de Datos de Capacitación

DIRECCIÓN

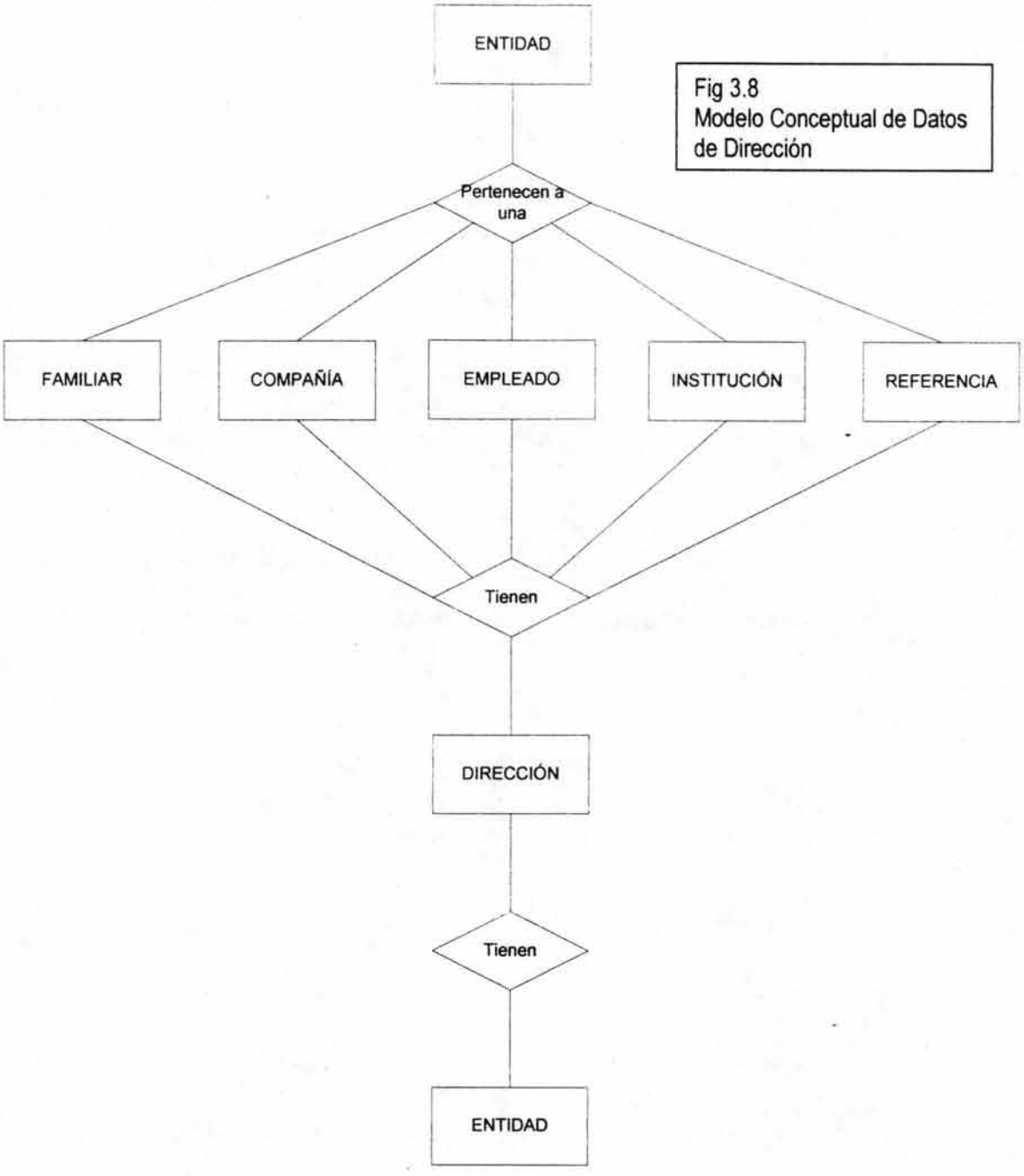


Fig 3.8
Modelo Conceptual de Datos de Dirección

Cada entidad expuesta cuenta con una serie de atributos de cierto tipo cada uno de ellos, estos pueden visualizarse en el archivo de Microsoft Visio ya mencionado. Por lo tanto ya prácticamente tenemos nuestra definición de datos para nuestro sistema.

3.1.2 Diseño Físico

Ahora utilizaré Power Designer 6.1 para realizar el Diagrama de Base de Datos, en base al Modelo Conceptual expuesto y al modelo entidad relación mencionado en el primer capítulo.

Es conveniente que cuando se realiza el modelo físico de la BD, este se segmente en módulos propios de los que será el sistema. Por lo tanto veremos en los siguientes diagramas que cada módulo del sistema de RH se compone de cierto número de tablas. Este diagrama se encuentra en un archivo PDM, propio de Power Designer.

Del modelo conceptual, no necesariamente se realiza una tabla por cada entidad, si no que conforme analizamos nuestro MCD nos daremos cuenta que él mismo nos indica cuáles entidades serán tablas, cuáles no y cuáles de las relaciones lo serán, ya que estas últimas juegan un papel muy importante en lo que será nuestra consistencia de datos una vez hecha la Base de Datos.

Existe en especial una tabla principal llamada RH_EMPLEADO, la cual traducimos a partir de la entidad EMPLEADO, es por eso que será la primera en exponerse, de esta parte en varias tablas del sistema, ya que la clave del empleado en nuestro caso será la llave principal. En la Fig. 3.9 podremos apreciar la tabla principal de nuestra BD así como el diagrama completo, el cual se encuentra en el archivo Modelo Físico.PDM y en el archivo RECURSOS_HUMANOS.RTF se encuentra un informe[39] detallado de la base de datos, incluyendo todos los objetos y las descripciones de cada uno de ellos.

RH_EMPLEADO			
INT_COD_EMPLEADO	<pk>	int	not null
SMI_COD_IDENTIFICADOR_NOMINA	<fk>	smallint	not null
STR_COD_NOMINA	<fk>	char(2)	not null
SMICOD_TIPOPAGO	<fk>	smallint	not null
SMI_COD_TIPO_EMPLEADO	<fk>	smallint	not null
SMI_COD_EDO_CIVIL	<fk>	smallint	not null
STR_APPATERNO		varchar(30)	not null
STR_APMATERNO		varchar(30)	not null
STR_NOMBRE		varchar(30)	not null
STR_HORARIO_ENTRADA		varchar(5)	null
STR_HORARIO_SALIDA		varchar(5)	null
DAT_FECHA_MATRIMONIO		datetime	null
DAT_FECHA_NACIMIENTO		datetime	not null
STR_LUGAR_NACIMIENTO		varchar(60)	not null
BOL_VIGENTE		bit	not null
IMG_FOTOGRAFÍA		image	null
STR_NACIONALIDAD		varchar(30)	not null
STR_SEXO		char(1)	not null
BOL_FIRMA		bit	not null
IX_NOMBRE_EMPLEADO			

Figura 3.9
Tabla RH_EMPLEADO

[39] El informe se puede generar desde Power designer.

No presentaré en este documento todas las tablas del sistema, esta podrán verlas con lujo de detalle en el archivo PDM, aqui solo presentaré las mas importantes.

RH_EMPLEADO, como podemos observar tiene una única llave primaria, a la cual harán referencia las tablas que se encuentran en el siguiente listado, los nombres que vemos después de los dos puntos, no son mas que los nombres de las referencias.

```
RH_ALUMNO_CURSO: FK_RH_ALUMN_PERTENECE_RH_EMPLE
RH_APOYO: FK_RH_APOYO_TIENE_APO_RH_EMPLE
RH_ASIGNACION_AGENCIA: FK_RH_ASIGN_AGENCIA_A_RH_EMPLE
RH_CAPACITACION: FK_RH_CAPAC_TIENE_CAP_RH_EMPLE
RH_CONOCIMIENTO_GRAL: FK_RH_COCOC_TIENE_CON_RH_EMPLE
RH_CONTRATO: FK_RH_CONTR_TIENE_CON_RH_EMPLE
RH_CREDITO_INFONAVIT: FK_RH_CREDI_TIENE_CRE_RH_EMPLE
RH_CUENTA_BANCARIA: FK_RH_CUENT_TIENE_CTA_RH_EMPLE
RH_DEUDA: FK_RH_DEUDA_TIENE_DEU_RH_EMPLE
RH_DIRECCION_EMPLEADO: FK_RH_DIREC_TIENE_DIR_RH_EMPLE
RH_ECONOMICO: FK_RH_ECONO_TIENE_ECO_RH_EMPLE
RH_ESCOLARIDAD: FK_RH_ESCOL_TIENE_ESC_RH_EMPLE
RH_ESTUDIOS_ACTUALES: FK_RH_ESTUD_TIENE_EST_RH_EMPLE
RH_EVALUACION: FK_RH_EVALU_TIENE_EVA_RH_EMPLE
RH_EXAMEN: FK_RH_EXAME_REALIZA_E_RH_EMPLE
RH_EXP_LABORAL: FK_RH_EXP_L_TIENE_EXP_RH_EMPLE
RH_FAMILIAR: FK_RH_FAMIL_TIENE_FAM_RH_EMPLE
RH_HISTORIAL_MEDICO: FK_RH_HISTO_REF_334_RH_EMPLE
RH_IDIOMA: FK_RH_IDIOM_ESTUDIA_I_RH_EMPLE
RH_IMPARTE_CURSO: FK_RH_IMPAR_IMPARTE_E_RH_EMPLE
RH_INCIDENCIA: FK_RH_INCID_TIENE_INC_RH_EMPLE
RH_LOCALIZACION: FK_RH_LOCAL_REF_803_RH_EMPLE
RH_MOD_MOVIMIENTO: FK_RH_MOD_M_REALIZA_M_RH_EMPLE
RH_MONTO_PTU: FK_RH_MONTO_REF_2123_RH_EMPLE
RH_MOVIMIENTO: FK_RH_MOVIM_TIENE_MOV_RH_EMPLE
RH_NOMINA_ESPECIAL: FK_RH_NOMIN_TIENE_N_E_RH_EMPLE
RH_PRESTAMO: FK_RH_PREST_SOLICITA__RH_EMPLE
RH_PUESTO_ASIGNADO: FK_PUESTO_CONTIENE_EMPLEADO
RH_REFERENCIA: FK_RH_REFER_TIENE_REF_RH_EMPLE
RH_REGISTRO: FK_RH_REGIS_TIENE_REG_RH_EMPLE
RH_SALARIO: FK_RH_SALAR_TIENE_SAL_RH_EMPLE
RH_SOLICITUD_ASPIRANTE: FK_RH_SOLIC_TIENE_SOL_RH_EMPLE
RH_TOTAL_MOVIMIENTO: FK_RH_TOTAL_TIENE_TOT_RH_EMPLE
RH_ZONA_ECONOMICA: FK_RH_ZONA__PERTENECE_RH_EMPLE
SYS_USUARIO: FK_SYS_USUA_PERTENECE_RH_EMPLE
```

Por otro lado, esta tabla cuenta con llaves foráneas, es decir, esta tabla también hace referencia a otras tablas:

```
SMICOD_TIPOPAGO
REFERENCES RH_TIPO_PAGO (SMICOD_TIPOPAGO)
SMI_COD_IDENTIFICADOR_NOMINA, STR_COD_NOMINA
REFERENCES RH_NOMINA (SMI_COD_IDENTIFICADOR_NOMINA, STR_COD_NOMINA)
SMI_COD_TIPO_EMPLEADO
REFERENCES RH_TIPO_EMPLEADO (SMI_COD_TIPO_EMPLEADO)
SMI_COD_EDO_CIVIL
REFERENCES RH_EDO_CIVIL (SMI_COD_EDO_CIVIL)
```

Y por último esta tabla cuenta con dos índices, uno pues es la llave primaria, y el otro se compone de tres campos:

```
PK_RH_EMPLEADO          INT_COD_EMPLEADO

IX_NOMBRE_EMPLEADO      INT_COD_EMPLEADO STR_APPATERNO, STR_APMATERNO,
STR_NOMBRE
```

Las tablas que veremos a continuación , se encargarán de almacenar las fórmulas creadas por los usuarios, las cuales representarán a cada concepto de nómina calculado en el proceso de cálculo de nómina, es por eso que la tabla RH_CONCEPTO tiene hacia ella varias referencias, ya que a partir de aquí se generan los movimientos y los diferentes rubros del módulo de Nómina. Entonces veamos esta tabla en la Fig. 3.10.

RH_CONCEPTO

SMI_COD_CONCEPTO	<pk>	smallint	not null
STR_CLAVE_CONCEPTO		char(4)	not null
SMI_COD_TIPO_CONCEPTO	<fk>	smallint	not null
SMI_COD_IDENTIFICADOR_NOMINA	<fk>	smallint	not null
STR_COD_NOMINA	<fk>	char(2)	not null
BOL_IMPRESO		bit	not null
STR_DESCRIPCION		varchar(60)	not null
BOL_IMSS		bit	not null
BOL_SALDOS		bit	not null
BOL_INCIDENCIA		bit	not null
SMI_COD_BASEISPT	<fk>	smallint	null

Figura 3.10
Tabla RH_CONCEPTO

RH_CONCEPTO también tiene una única llave primaria a la cual hacen referencia las siguientes tablas:

```
RH_BITACORA_CALCULO: FK_RH_BITAC_PERTENECE_RH_CONCE
RH_CONCEPTO_TEMPLEADO: FK_RH_CONCE_CONTIENE__RH_CONCE
RH_FORMULA: FK_RH_FORMU_TIENE_FOR_RH_CONCE
RH_INCIDENCIA: FK_RH_INCID_TIENEC_RH_CONCE
RH_MOVIMIENTO: FK_RH_MOVIM_PERTENECE_RH_CONCE
RH_OPERANDO_MOV_ACUMULADO: FK_RH_OPERA_OPERANDO__RH_CONCE
RH_PRIORIDAD_DESC: FK_RH_PRIOR_TIENE_PRI_RH_CONCE
RH_TOPE: FK_RH_TOPE_TIENE_TOP_RH_CONCE
```

RH_CONCEPTO hace referencia las tablas:

```
SMI_COD_IDENTIFICADOR_NOMINA, STR_COD_NOMINA
REFERENCES RH_NOMINA (SMI_COD_IDENTIFICADOR_NOMINA, STR_COD_NOMINA)
SMI_COD_TIPO_CONCEPTO
REFERENCES RH_TIPO_CONCEPTO (SMI_COD_TIPO_CONCEPTO)
SMI_COD_BASEISPT
REFERENCES RH_BASEISPT (SMI_COD_BASEISPT)
```

Ahora veamos este segmento de la base de datos, que, como ya mencioné, es muy importante en el proceso de cálculo de nómina. Veámoslo en la Fig. 3.11.

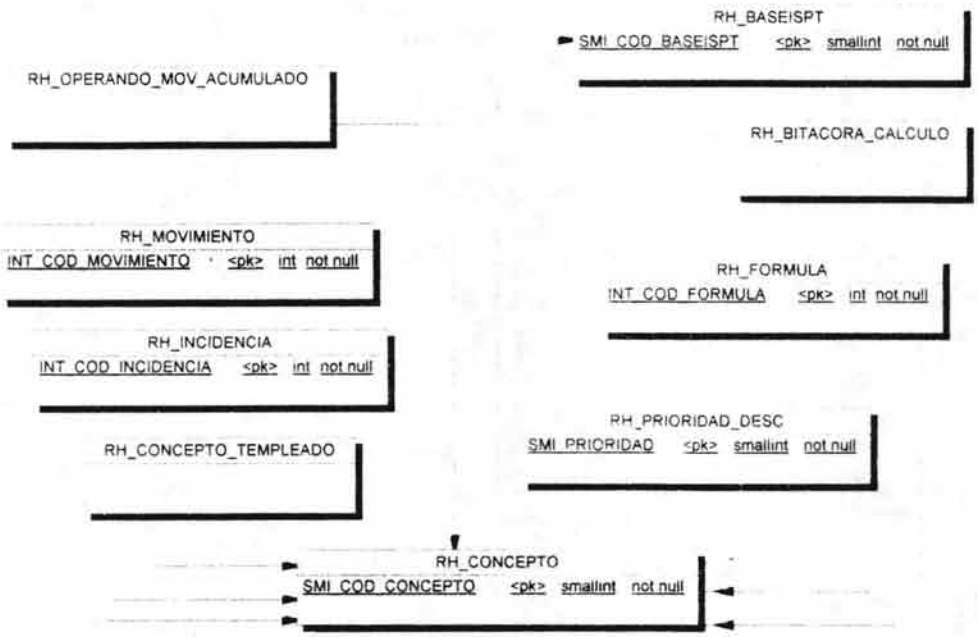


Figura 3.11
Tabla RH_CONCEPTO

Ahora veamos la tabla RH_NÓMINA, esta contendrá las nóminas existentes y pertenecientes a todas las compañías creadas. Veámosla en la Fig 3.12

RH_NOMINA			
SMI_COD_COMPAÑIA	<fk>	smallint	not null
SMI_COD_IDENTIFICADOR_NOMINA	<pk>	smallint	not null
STR_COD_NOMINA	<pk>	char(2)	not null
SMI_COD_TIPO_NOMINA	<fk>	smallint	null
STR_DESCRIPCION		varchar(60)	not null

Figura 3.12
Tabla RH_NÓMINA

RH_NÓMINA cuenta con dos llaves primarias, las cuales son la clave numérica de la nómina y la clave del tipo cadena de la misma, esta última es la que mas utilizan los usuarios y las tablas que hacen referencia a esta son:

```
RH_AGUINALDO: FK_RH_AGUIN_PERTENECE_RH_NOMIN
RH_CONCEPTO: FK_RH_CONCE_PERTENECE_RH_NOMIN
RH_EMPLEADO: FK_RH_EMPLE_PERTENECE_RH_NOMIN
RH_NOMINA_ESPECIAL: FK_RH_NOMIN_TIENE_NOM_RH_NOMIN
RH_OPERANDO_MOV_ACUMULADO: FK_RH_OPERA_OPERANDO__RH_NOMIN
RH_PERIODO: FK_RH_PERIO_CONTIENE__RH_NOMIN
RH_PTU: FK_RH_PTU_PERTENECE_RH_NOMIN
RH_VACACIONES: FK_RH_VACAC_CONTIENE__RH_NOMIN
```

RH_NÓMINA hace referencia a las siguientes tablas:

```
SMI_COD_COMPANIA
REFERENCES RH_COMPANIA (SMI_COD_COMPANIA)
SMI_COD_TIPO_NOMINA
REFERENCES RH_TIPO_NOMINA (SMI_COD_TIPO_NOMINA)
```

Veamos este segmento de la base de datos que se encarga de gestionar todo lo referente a la nómina en la Fig. 3.13.

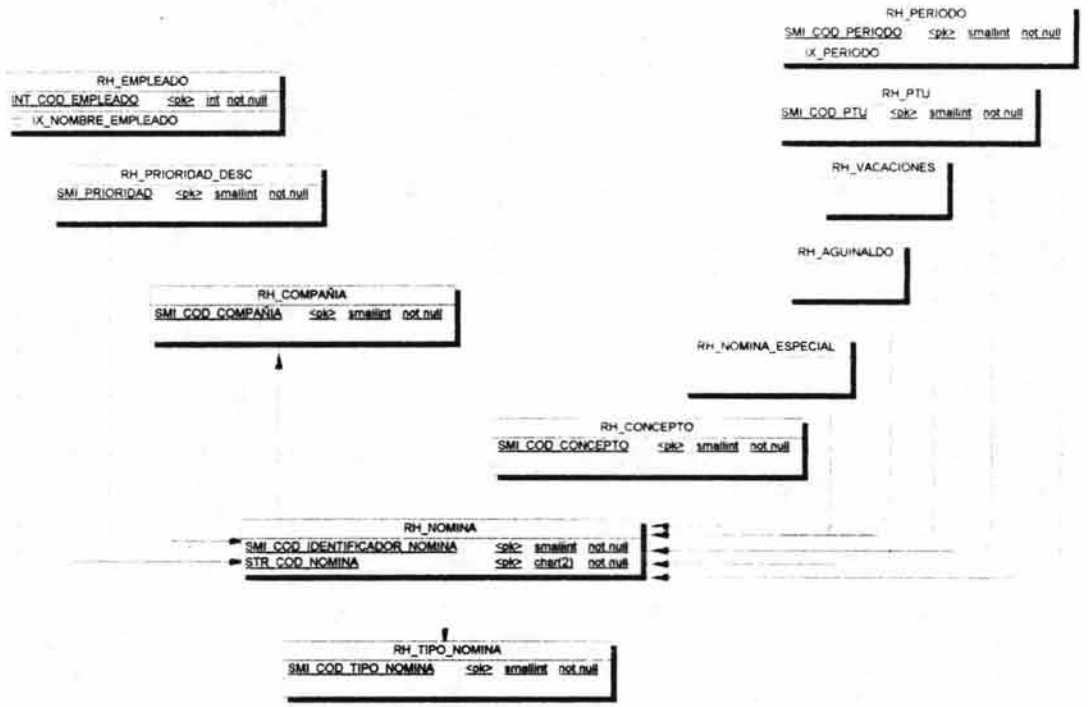


Figura 3.13
Módulo de Nómina

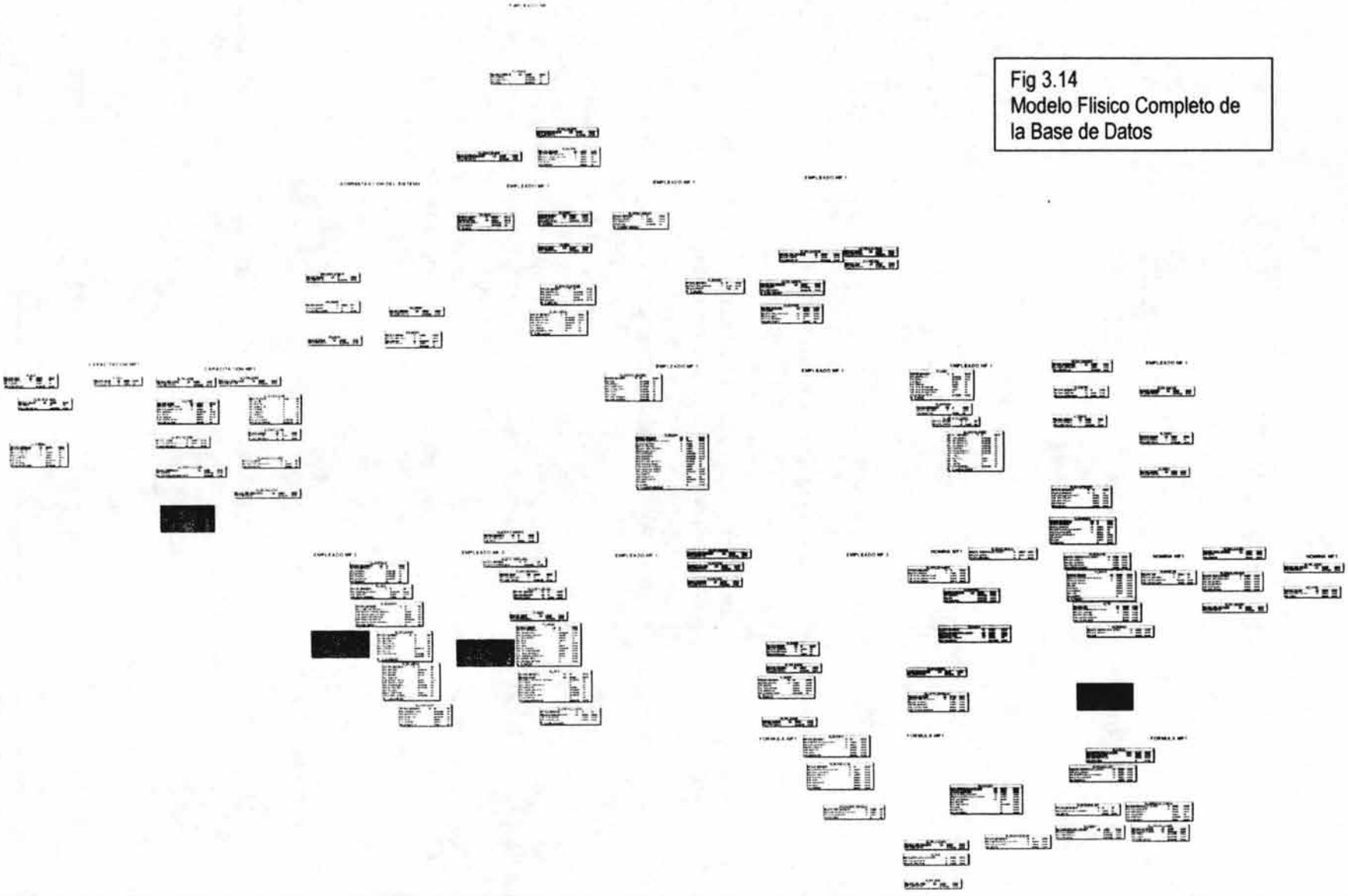


Fig 3.14
Modelo Flisico Completo de la Base de Datos

3.2 Solución a los procesos

3.2.1 Modelado de Objetos

En este apartado podremos aplicar lo que vimos en el capítulo I, utilicé UML para el modelado de objetos. Comencemos por todo lo necesario para poder acceder a la base de datos.

El esquema que se utilizará para la interacción de la Interfaz de usuario, los procesos y la Base de datos, se encuentra en las siguientes clases:

- CONEXION. Esta clase se encarga de comunicarse con la BD por medio de JDBC, esta es independiente, es decir, no hereda de nadie mas.
- TRANSACCION. Hereda de la clase CONEXIÓN, y es puente entre las clases dónde se almacenan las reglas de negocio y el acceso a la BD.
- CLASE_MODULO_SISTEMA. Esta clase representa a todas aquellas clases dónde se ensamblan las sentencias SQL, esta hereda de TRANSACCION para mandar a ejecutar dichas sentencias.
- CClaseServlet. Esta clase en el diagrama representa todas las clases que ejecutarán o mandarán a ejecutar los procesos que definen las reglas de negocio, y está íntimamente relacionada con CLASE_MODULO_SISTEMA.
- CclaseApplet. Se encarga de dibujar toda la Interfaz de usuario de su módulo correspondiente, así mismo, manda a ejecutar la clase CclaseServlet correspondiente.
- MENSAJE_ERR. Esta clase no hace otra cosa mas que gestionar los mensajes de error si estos existieran en alguna de las otras clases, es por eso que la gran mayoria de las clases se relacionan con esta.

En la Fig. 3.15 se muestra el diagrama hecho en Rational Rose de las clases expuestas. El diagrama se encuentra en el archivo Acceso_BD.mdl

Fig. 3.15 Modelado de objetos del esquema básico de clases y Acceso de a la BD

En base al esquema anterior, haremos todos los módulos del sistema, comenzaremos por el módulo principal, de dónde se partirá todo el sistema.

CSRHApplet. Esta clase es un Applet, y es el applet principal de todos el sistema, apartir de aquí, todas las demás applet's heredaran sus características, esta clase solamente tratará Interfaz. Esta clase hereda de la clase JApplet, por lo tanto, presentaré la primer clase que comenzará a resolver problemas del módulo Inventario de Recursos Humanos.

3.3 Diseño del Módulo Inventario de Recursos Humanos

CinventarioApplet. Este applet que se encargará de toda la interfaz del Inventario de Recursos Humanos, hereda de CSRHApplet y esta heredará de la clase CseguridadSistema quea su vez heredará sus caracteristicas a sus clases inferiores que son:

- CSRHApplet
- CDatoPersonalApplet
- CEscolaridadCapacitacionApplet
- CDatoApoyoApplet
- CDatoEconomicoApplet
- CContratacionApplet

Estas se encargarán de gestionar la GUI [40], y en base al esquema básico de la Fig. 3.15 mostraré el Modelado de Objetos del módulo de Inventario de Recursos Humanos en la Fig. 3.16

[40] Graphic User Interfaz .- Interfaz gráfica de usuario, en este caso con Components Swing.

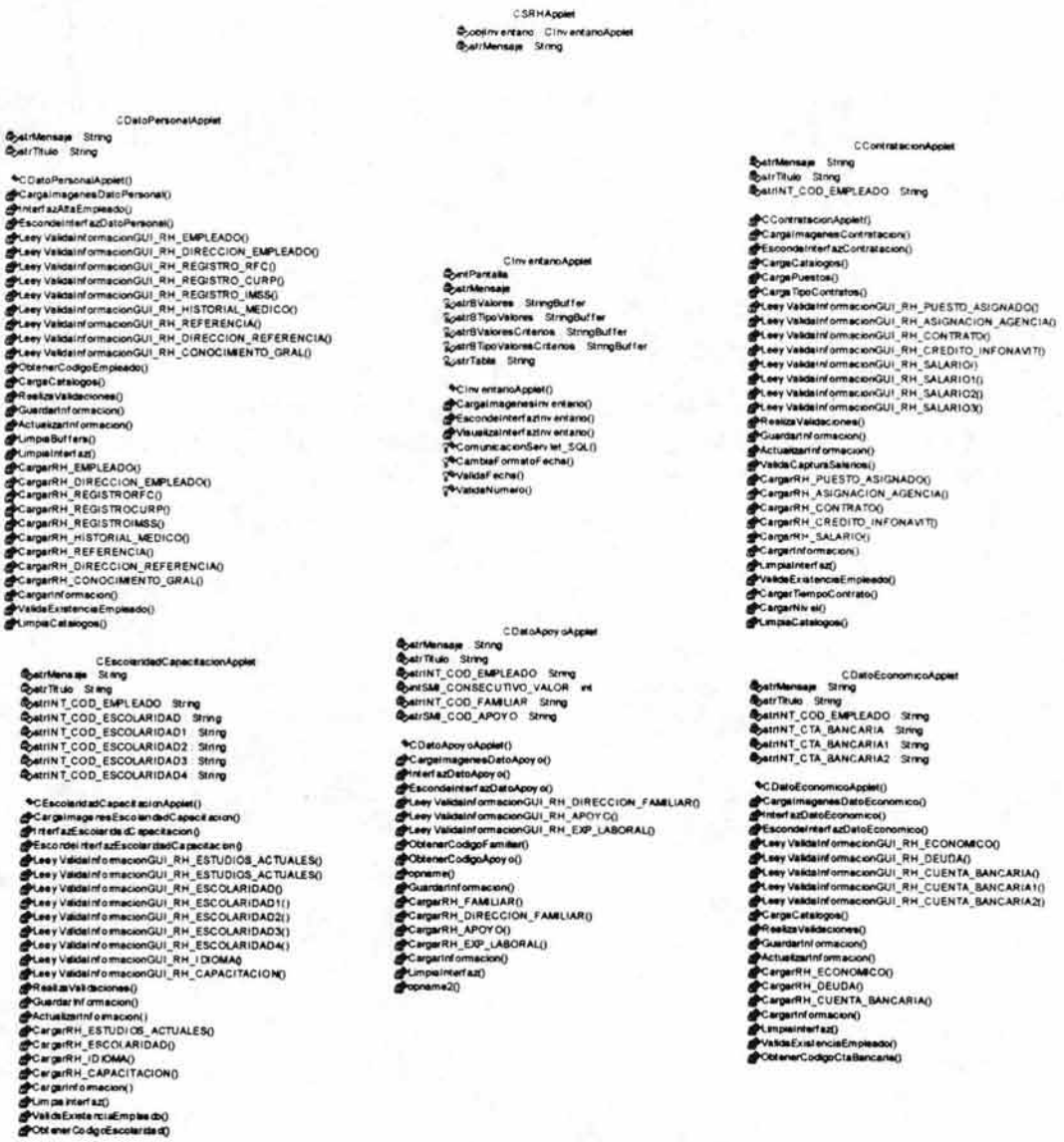


Figura 3.16
Modelado de objetos del Inventario de RH.

Ahora bien, CinventarioApplet se relaciona por medio de una comunicación especial llamada HTTP tunneling (la cual se explicará en el siguiente capítulo) con la clase CinventarioServlet la cual se encargará

de recibir información proveniente de CInventarioApplet, procesarla y canalizarla a la siguiente clase llamada CINVENTARIO, la cual posee todos aquellos métodos específicos para el acceso a la base de datos, sin embargo esta no se comunica con la Base de Datos directamente si no que hereda las características de la clase TRANSACCIÓN, esta se encarga de clasificar el tipo de acceso a la Base de Datos, de tal manera que hereda las características de la clase CONEXIÓN, esta última se encarga de conectarse a la Base de datos directamente. Veamos el modelado de objetos en la Fig. 3.17.

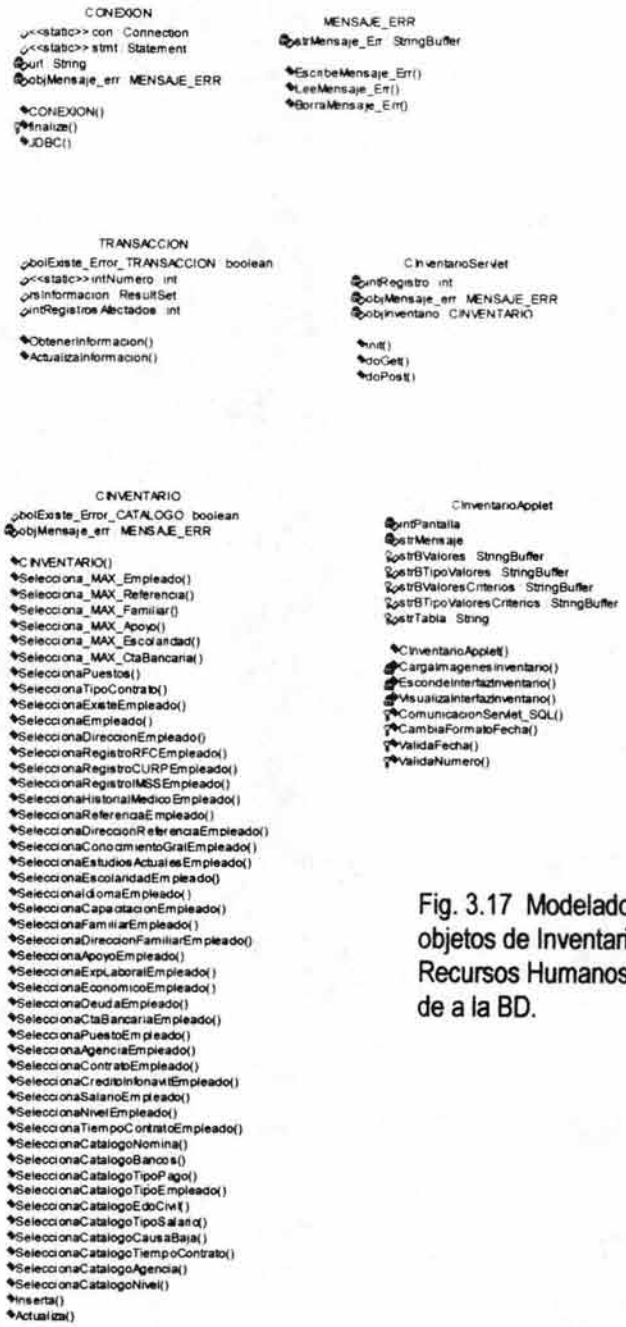


Fig. 3.17 Modelado de objetos de Inventario de Recursos Humanos y Acceso de a la BD.

La clase MENSAJE_ERR se encargará de gestionar los mensajes de error en todo el sistema, es por eso que veremos objetos miembro de esta clase en todos los módulos.

Hablando mas específicamente, del lado del servidor, es decir a partir del servlet, en este caso CInventarioservlet, este recibe mensajes por medio de la comunicación HTTP tunneling provenientes del applet, de estos mensajes depende el comportamiento del servlet, este a su vez crea dos instancias de las clases CINVENTARIO y MENSAJE_ERR. CINVENTARIO hereda directamente de TRANSACCIÓN como lo vimos en la figura anterior, y envía mensajes a esta última. TRANSACCIÓN recibe los mensajes de CINVENTARIO y dependiendo de estos mensajes es el comportamiento de TRANSACCIÓN, el comportamiento especifico se explicará en el siguiente capítulo. TRANSACCION hereda directamente de la clase CONEXIÓN, esta vendría siendo nuestra súper clase. Veamos el diagrama de interacción de todo este apartado en la Fig. 3.17.

Fig. 3.17 Diagrama de Interacción del Módulo de Recursos Humanos

3.4Diseño de comunicación del Usuario con el Sistema – Casos de Uso

De acuerdo con el análisis realizado en el capítulo anterior, se diseñará cómo el usuario navegará en los diferentes apartados del sistema, esto lo haremos mediante casos de uso, los cuales se encuentran en el archivo SRH.mdl de Rational Rose. Veamos primero en la Figura 3.18 la organización de los módulos del Sistema.

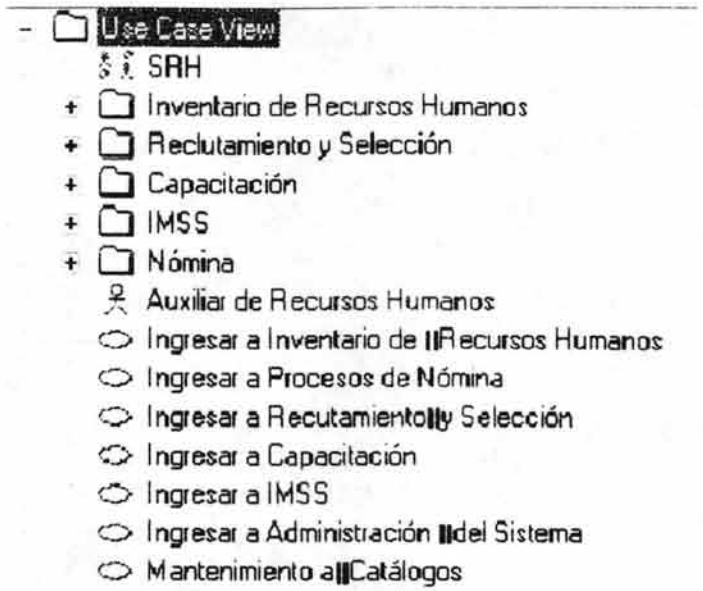


Figura 3.18
Organización General del Sistema

En las siguientes figuras veremos la expansión de cada ramificación.

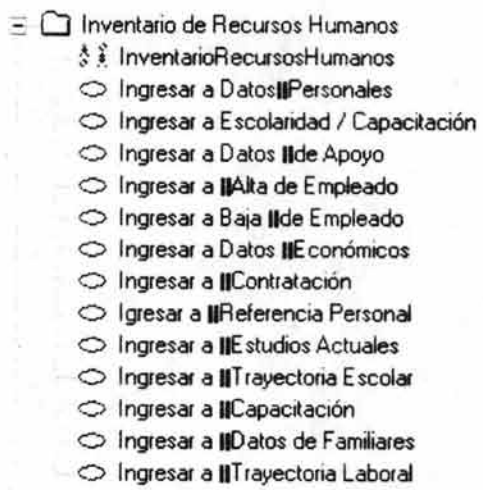


Figura 3.19
Organización del Módulo de Inventario de Recursos Humanos

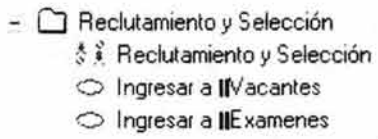


Figura 3.20
Organización del Módulo de Reclutamiento y Selección

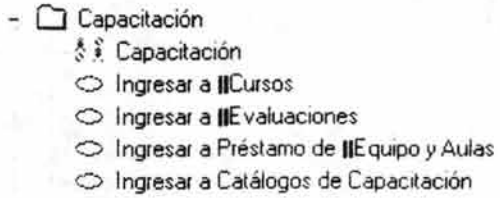


Figura 3.21
Organización del Módulo de Capacitación

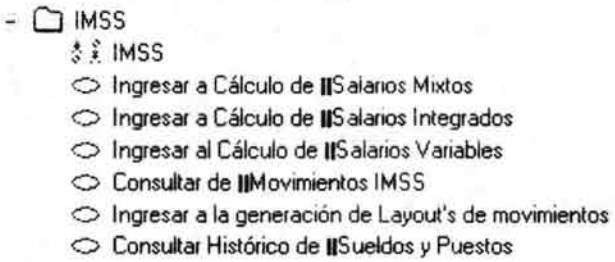


Figura 3.22
Organización del Módulo de IMSS

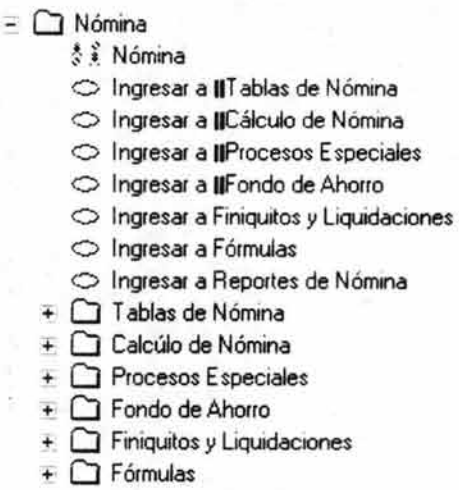


Figura 3.23
Organización del Módulo de Nómina

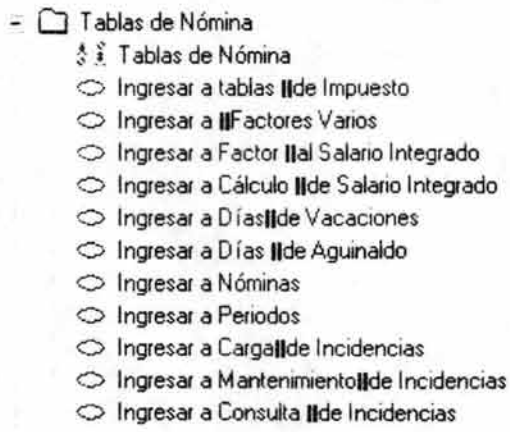


Figura 3.24
Organización del Submódulo de Tablas de Nómina

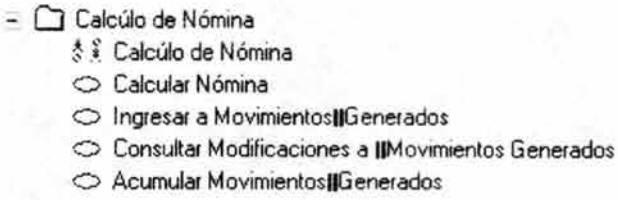


Figura 3.25
Organización del Submódulo de Cálculo de Nómina

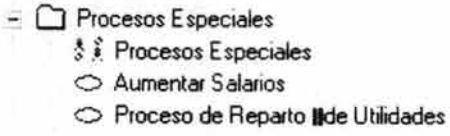


Figura 3.26
Organización del Submódulo de Cálculo de Nómina

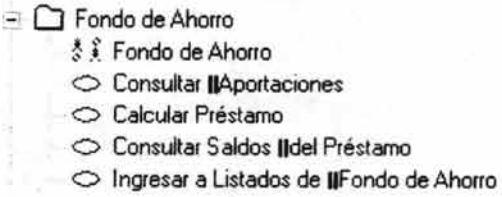


Figura 3.27
Organización del Submódulo de Fondo de Ahorro

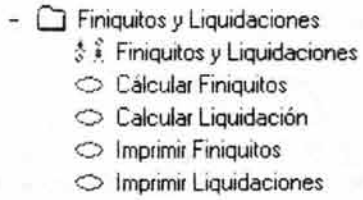


Figura 3.28
Organización del Submódulo de Finiquitos y Liquidaciones

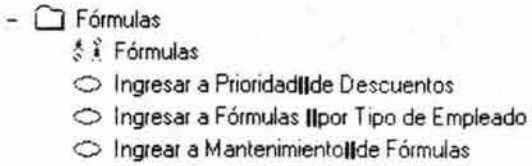


Figura 3.29
Organización del Submódulo de Fórmulas

Ahora veamos los casos de uso de cada paquete en las siguientes figuras. Los casos de uso que veremos a continuación, es el diseño de navegación del usuario en el sistema, es decir, cada caso de uso representa de qué manera el usuario va a ingresar en los módulos y submódulos del Sistema.

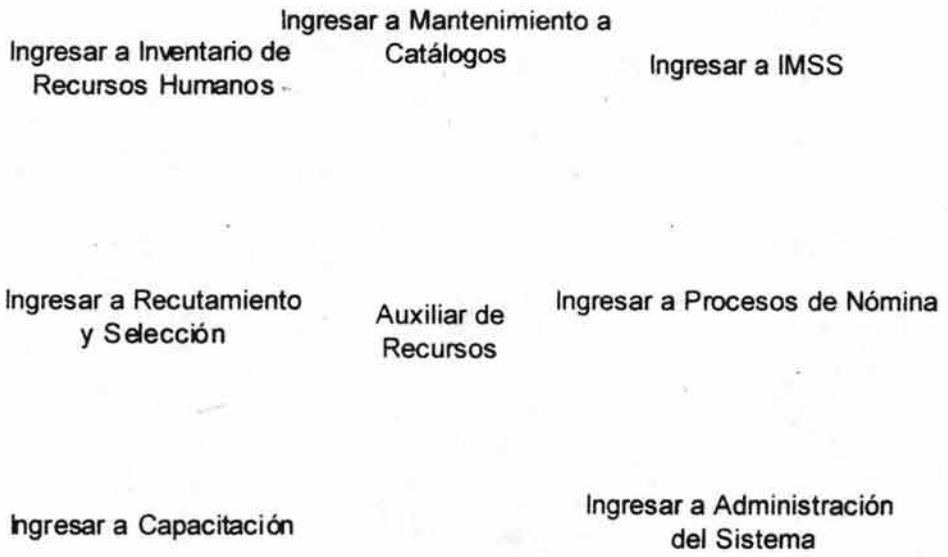


Figura 3.30
Caso de Uso del Sistema de Recursos Humanos

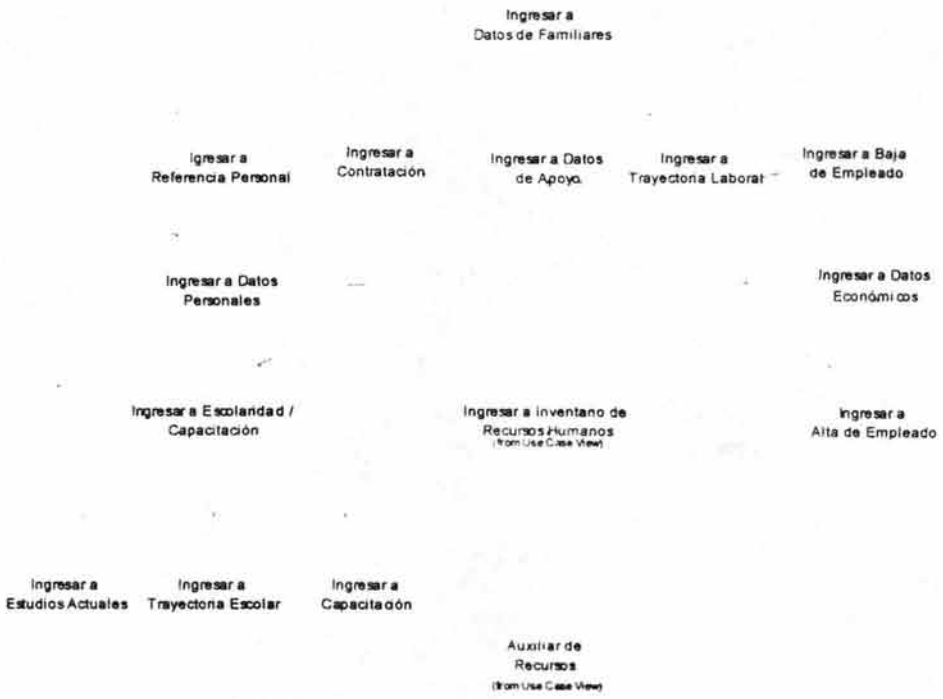


Figura 3.31
Caso de Uso de Inventario de Recursos Humanos

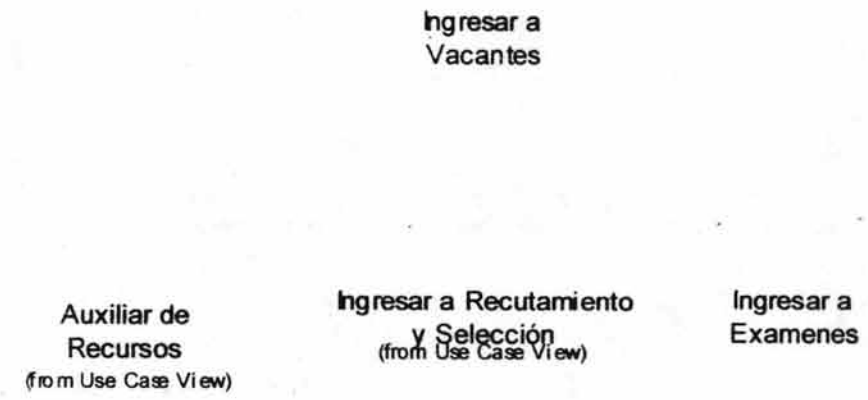


Figura 3.32
Caso de uso de Reclutamiento y Selección

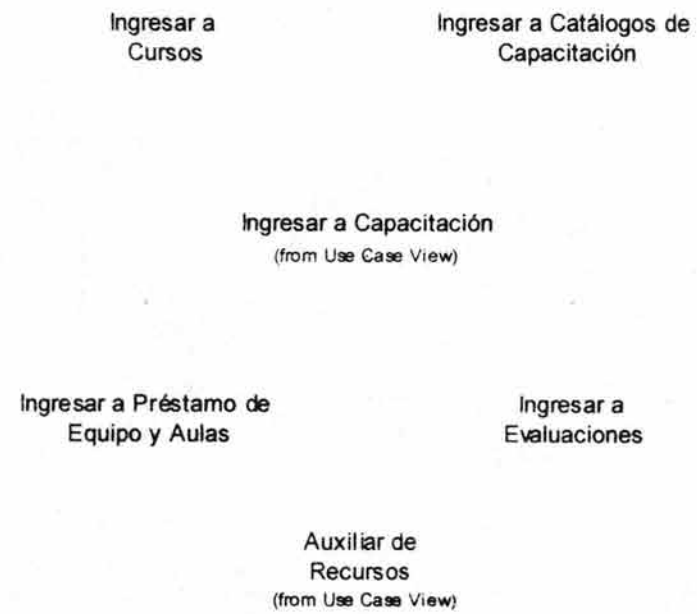


Figura 3.33
Caso de uso de Capacitación

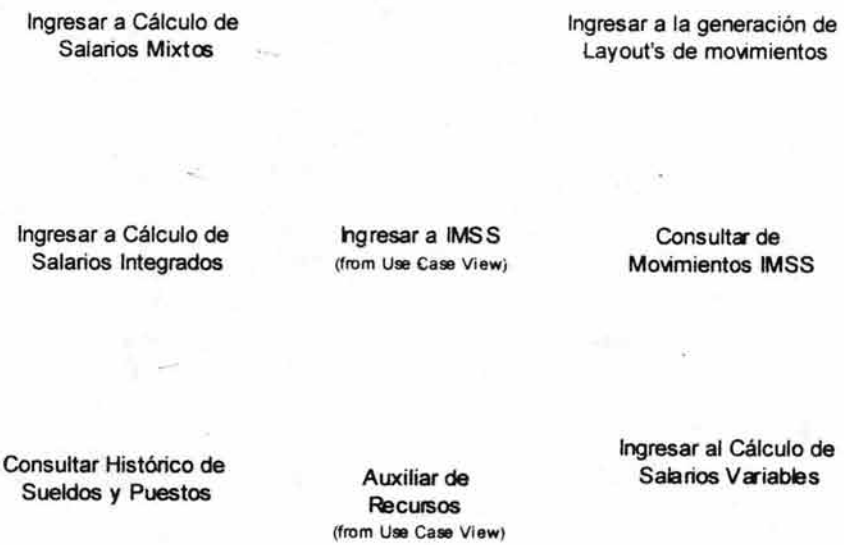


Figura 3.34 Caso de uso de IMSS

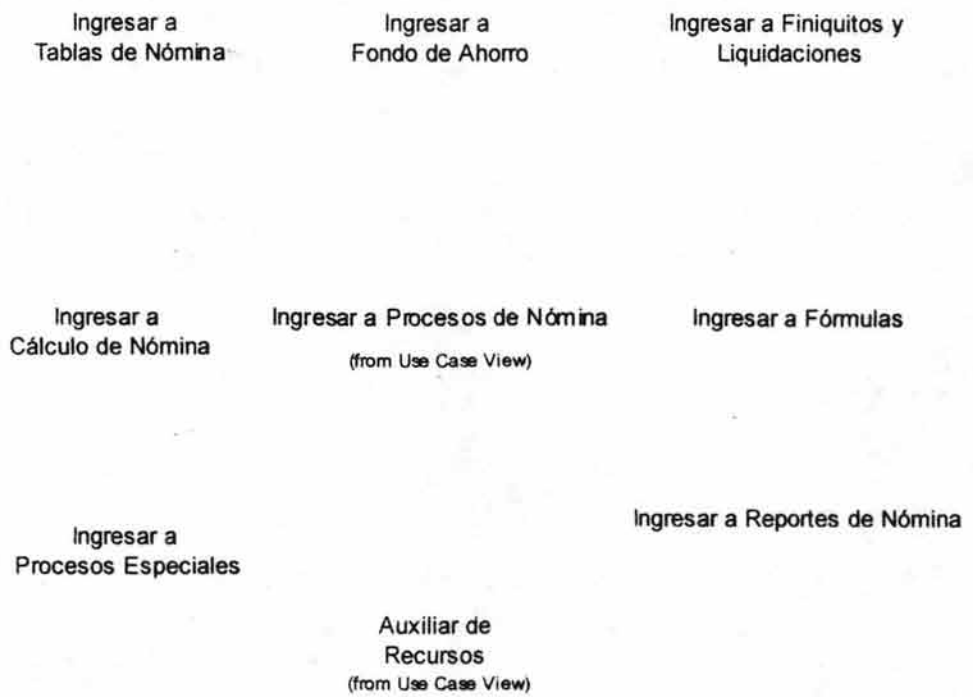


Figura 3.35
Caso de uso de Procesos de Nómina

Figura 3.36
Caso de uso de Tablas de Nómina

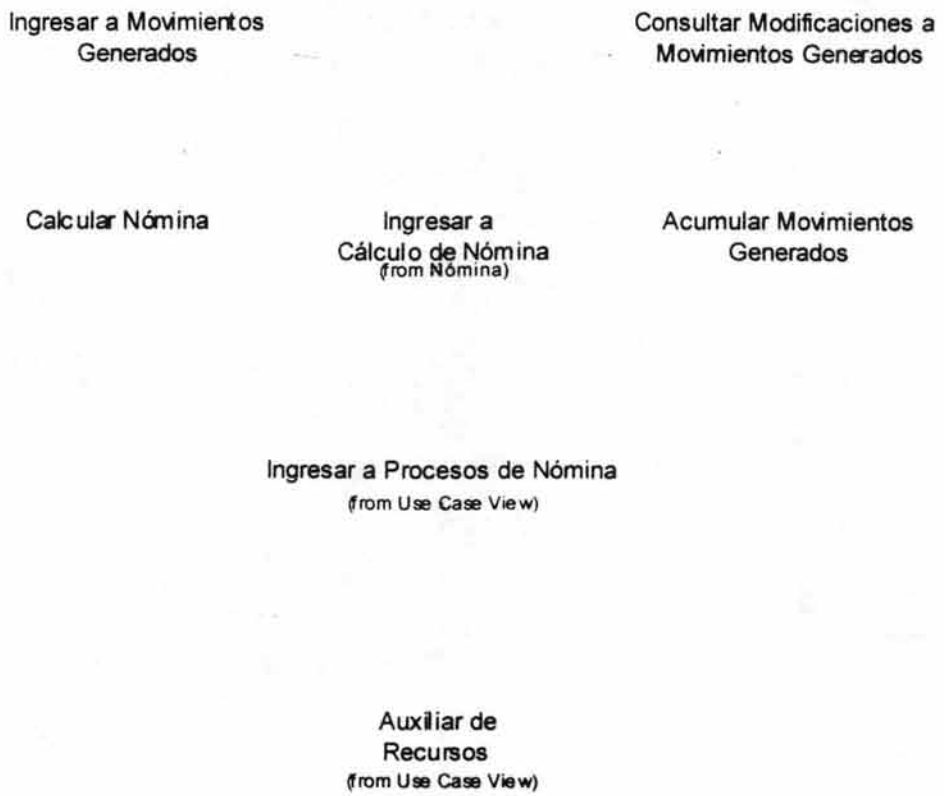


Figura 3.37
Caso de uso de Cálculo de Nómina

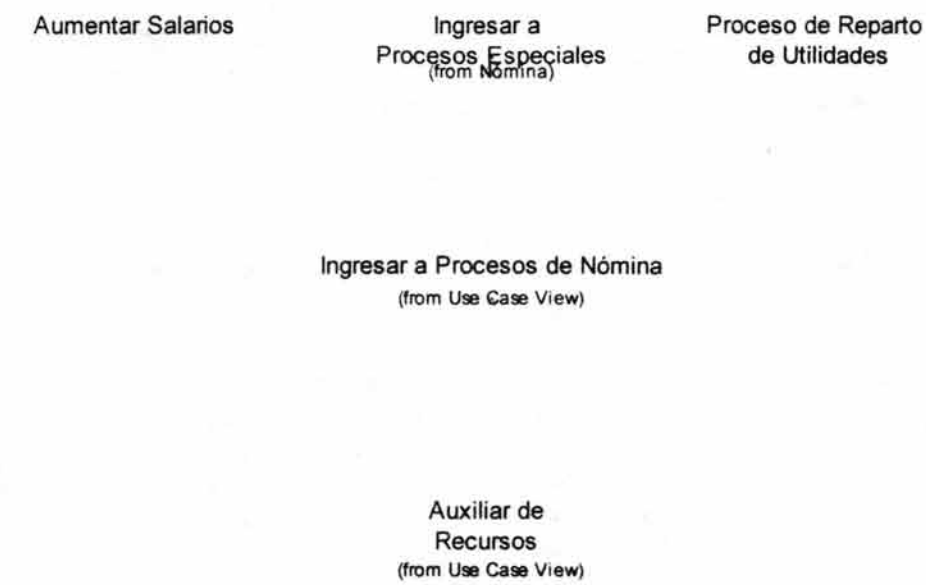


Figura 3.38
Caso de uso de Procesos Especiales

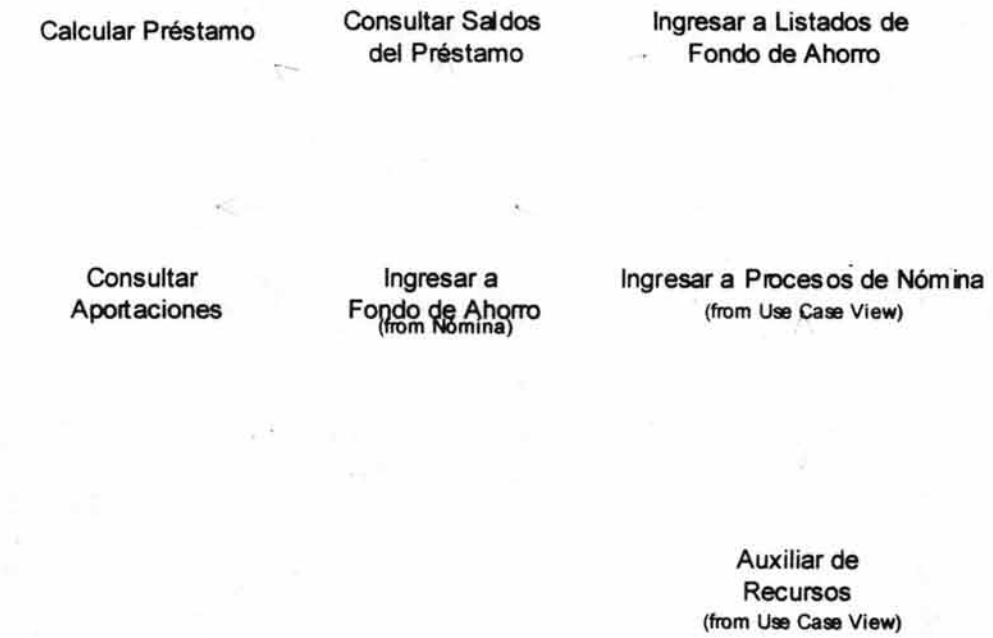


Figura 3.39
Caso de uso de Fondo de Ahorro

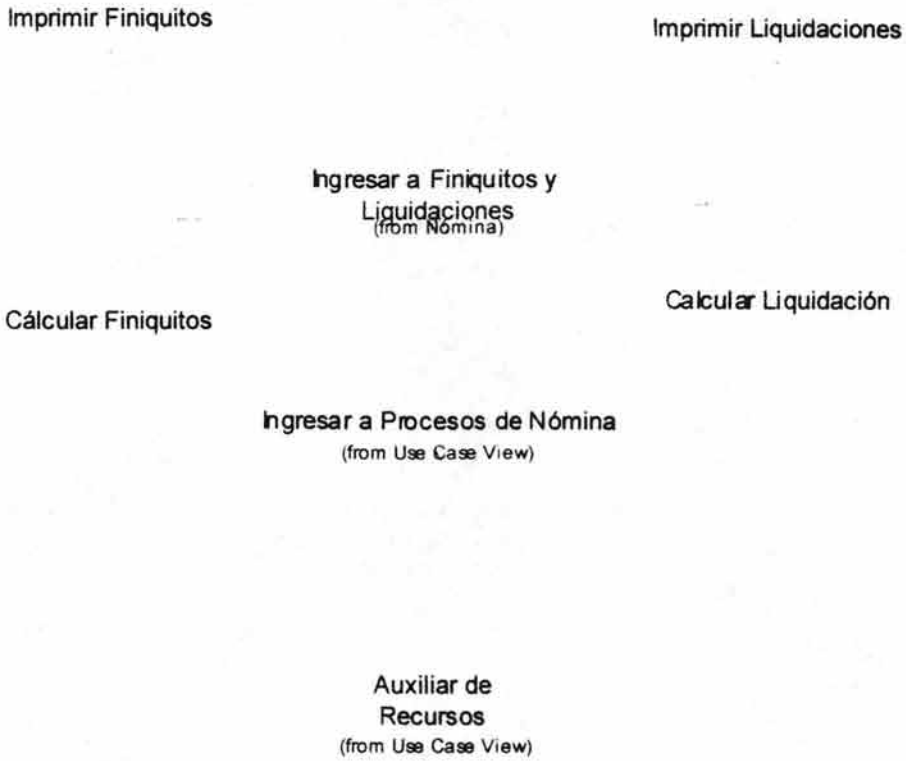


Figura 40
Caso de uso de Fondo de Ahorro

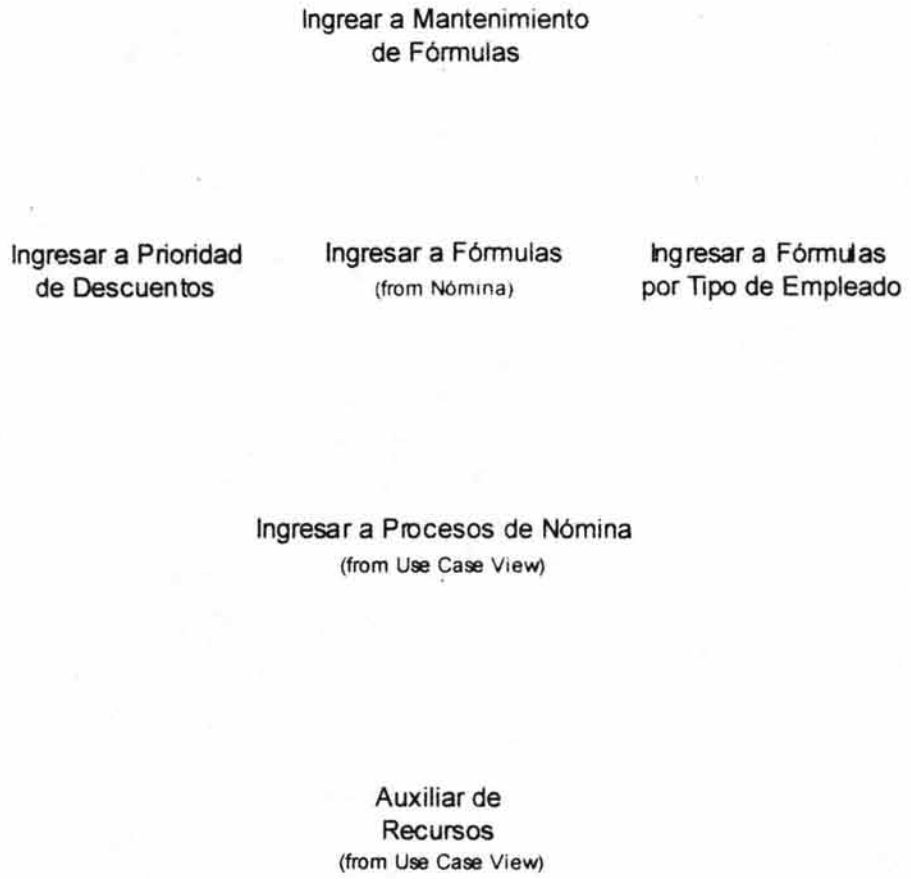


Figura 3.41
Caso de uso de Fórmulas

Capítulo IV Desarrollo del Sistema de Recursos Humanos

4.1 Estándares

4.1.1 Base de datos

Para la Base de Datos utilicé los siguientes estándares:

Puesto que la Base de Datos se llama RECURSOS_HUMANOS:

- Todas las tablas tendrán el prefijo RH, sus nombres serán en mayúsculas, estos últimos serán lo más descriptivos posibles a los registros que almacenen.
- Cada campo de cada tabla tendrán como prefijo tres caracteres que los identificará de qué tipo de datos se trata:

Prefijo	Tipo de Dato
INT	Entero
SMI	Smallint
DEC	Flotante
DAT	Fecha
BOL	Boleano
IMG	Imagen
STR	Cadena de caracteres

- Se utilizará antes del nombre de cada campo y después del prefijo del tipo de dato y solo si es necesario, un prefijo mas que indicará si el código o clave.

Prefijo	Descripción
COD	Código (sólo numérico)
CVE	Clave (puede ser alfanumérico)
SEC	Consecutivo (sólo numérico)

* Entre los prefijos y los nombres de los campos, existirá un separador = _.

4.1.2 Programación

El sistema se encontrará programado en Java, asi que los estándares que se utilizarán son:

Prefijo	Elemento al que se le aplica el prerfijo
Int	Dato entero
dbl	Double
obj	Objeto miembro de una clase
C	Nombre de Clase
bol	Booleano
str	String
strB	StringBuffer
vector	Vector
enum	Enumeration
URL	URL
strTk	StringTokenizer

Sufijo	Elemento al que se le aplica el sufijo
Applet	Cuando la clase se trata de un Applet
Servlet	Cuando la clase se trata de un Servlet

4.2 Desarrollo del Módulo Inventario de Recursos Humanos y Esquema Básico del Desarrollo del Sistema

Vamos a retomar a lo que se presentó en el capítulo anterior, solo que ahora veremos el desarrollo de este módulo en código fuente JAVA, sin presentar todas las líneas de código, ya que estas incrementarían en más del 1000% la robustez de la presente documentación, sin embargo presentaré lo más importante, comenzando por el esquema básico de clases que involucra a la conexión con la base de datos y la comunicación con la GUI.

Recordando del capítulo anterior nuestra superclase CONEXIÓN (Ver Fig. 4.1) que es la que se encarga de realizar la conexión a la Base de Datos por medio de JDBC, veremos ahora por fin cómo es que se hace. Veamos el código completo de la clase (Ver Fig. 4.2) y posteriormente detallaremos dicho código.

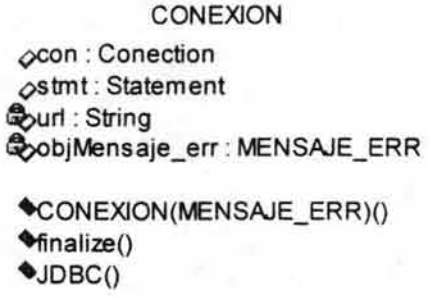


Figura 4.1
Clase CONEXIÓN

```
/*****************************************************************
        Creación:20 - Marzo - 2002
        Clase CONEXION
        Se encargará de realizar la conexión a la Base de datos con JDBC
                        Edgar Herrera Casas
                        Tesis:Análisis, Diseño y Desarrollo del Sistema de Recursos Humanos
                        para La Sección Amarilla
*****************************************************************/
import java.sql.*;
import java.awt.* ;
public abstract class CONEXION
{
        public static Connection con;
        public static Statement stmt;
        private String url;
        private static MENSAJE_ERR objMensaje_err;
        /*************************************************************
        Constructor: Tipo de driver y nombre del DSN
        *************************************************************/
        public CONEXION(MENSAJE_ERR objMensaje_err_Global)
        {       objMensaje_err = (MENSAJE_ERR)objMensaje_err_Global;
                url = "jdbc:odbc:RECURSOS_HUMANOS";
                JDBC();
        }
        /*************************************************************
        Destructor
        *************************************************************/
        protected void finalize() throws Throwable
        {       stmt.close();con.close();
                stmt = null; con = null;
        }
        /*************************************************************
        Inicializa el Driver JDBC, y realiza la conexión a la BD
        *************************************************************/
        public void JDBC()
        {       try
                { Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                }
                catch(java.lang.ClassNotFoundException e)
                {
                        objMensaje_err.EscribeMensaje_Err("CONEXION.JDBC() " + e.getMessage());
                }
                try
                {       con = DriverManager.getConnection(url, "sa", "");  }
                catch(SQLException ex)
                { objMensaje_err.EscribeMensaje_Err("CONEXION.JDBC()" + ex.getMessage());   }
        }
}
```

Figura 4.2
Código fuente de la Clase CONEXION

El primer método CONEXIÓN() es el constructor de nuestra clase, recibe un objeto miembro de la clase MENSAJE_ERR, esto es para dar a conocer los errores si es que estos existen, también manda a llamar al método más importante de esta clase JDBC():

```
public void JDBC()
{       try
        {       Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        }
        catch(java.lang.ClassNotFoundException e)
        {
                objMensaje_err.EscribeMensaje_Err("CONEXION.JDBC() " + e.getMessage());
        }
        try
        {       con = DriverManager.getConnection(url, "sa", "");
        }
        catch(SQLException ex)
        { objMensaje_err.EscribeMensaje_Err("CONEXION.JDBC()" + ex.getMessage());
        }
}
```

En este método realiza en sí la conexión, indicamos qué driver se va utilizar con la sentencia:

```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

Y se crea un objeto Connection especificando el nombre del driver con url = "jdbc:odbc:RECURSOS_HUMANOS";

```
con = DriverManager.getConnection(url, "sa", "");
```

El siguiente método es finalize, no hacemos mas que cerrar los objetos creados y destruirlos.

```
protected void finalize() throws Throwable
{       stmt.close();con.close();
        stmt = null; con = null;
}
```

Ahora veamos la siguiente clase de acuerdo a nuestra jerarquía de clases; TRANSACCIÓN, esta clase hereda directamente de CONEXIÓN, así que tiene acceso a los elementos **con, stmt, JDBC() y CONEXIÓN(),** recordemos la definición de nuestra clase en la Fig. 4.3 y su código fuente en la Fig 4.4.

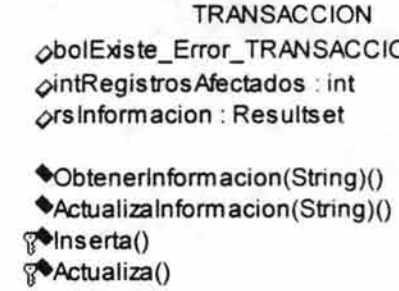


Figura 4.3

Clase TRANSACCION

```
/*************************************************************
        18 - Marzo - 2002
        SRH 1.0
        Última modificación: 11 - Febrero - 2003
        Edgar Herrera Casas
        Clase la cual se encarga de heredar de la clase CONEXION,
        para ejecutar cualquier sentencia SQL enviada por cualquier objeto del sistema
*************************************************************/
import java.sql.*;
import java.awt.* ;
public class TRANSACCION extends CONEXION
{       public boolean bolExiste_Error_TRANSACCION;
        public static int intNumero;
        public ResultSet rsInformacion;
        public int intRegistrosAfectados;
        private static MENSAJE_ERR objMensaje_err;
        /*************************************************************
         * Constructor: Llama al constructor de su Super Clase
         *************************************************************/
        public TRANSACCION(MENSAJE_ERR objMensaje_err_Global)
        {       super(objMensaje_err_Global);
                //El objeto que se recibe como parametro,
                //se almacenará en este otro objeto objMensaje_err
                objMensaje_err = (MENSAJE_ERR)objMensaje_err_Global;
                bolExiste_Error_TRANSACCION = false;
        }
        /*************************************************************
         * Ejecutará las sentencia SQL de selección de información
         *************************************************************/
        public void ObtenerInformacion(String strSentencia)
        {       try
                {       stmt = con.createStatement();
                        rsInformacion = stmt.executeQuery(strSentencia);
                }
                catch(SQLException ex)
                {       objMensaje_err.EscribeMensaje_Err("TRANSACCION.ObtenerInformacion(String
                        strSentencia)");
                }
        }
        /*************************************************************
         * Ejecutará las sentencia SQL de Inserción y actualización de información
         *************************************************************/
        public void ActualizaInformacion(String strSentencia)
        {       try
                {       stmt = con.createStatement();
                        System.out.println("Obteniendo Información.....un momento por favor");
                        intRegistrosAfectados = this.stmt.executeUpdate(strSentencia);
                        bolExiste_Error_TRANSACCION = false;
                }
                catch(SQLException ex)
                {       objMensaje_err.EscribeMensaje_Err("TRANSACCION.ActualizaInformacion(String strSentencia)
                        "  + ex.getMessage());
                        bolExiste_Error_TRANSACCION = true;
                        System.out.println("Ha ocurrido un error en TRANSACCION.ActualizaInformacion()"
                                                                + "SQLException: " + ex.getMessage());      }
        }
}
```

Figura 4.4
Código fuente de la Clase TRANSACCION

Veamos el primer método que es nuestro constructor de TRANSACCIÓN, este recibe como parámetro un objeto miembro de la clase MENSAJE_ERR, que como en todas las demás clases, nos servirá para dar conocimiento a la clase de dónde fue creado el primer objeto que dio pie a toda nuestra jerarquía de clases, y nuestro atributo bolExiste_Error_TRANSACCIÓN lo inicializamos en false ya que este nos indicará si existe error o no.

```
/*************************************************************
 * Constructor: Llama al constructor de su Super Clase
 *************************************************************/
public TRANSACCION(MENSAJE_ERR objMensaje_err_Global)
{
        super(objMensaje_err_Global);
        //El objeto que se recibe como parametro,
        //se almacenará en este otro objeto objMensaje_err
        objMensaje_err = (MENSAJE_ERR)objMensaje_err_Global;
        bolExiste_Error_TRANSACCION = false;
}
```

El siguiente método es ObtenerInformacion, este método lo utilizamos para obtener cualquier vista deseada, es decir, sentencias del tipo SELECT, ya que recibe la sentencia totalmente armada, esta es ejecutada por el método executeQuery() del objeto stmt, si hay algún error, se le notifica al objeto objMensaje:err, mandándole un mensaje por medio del método EscribeMensaje(). El resultado de la consulta, se guarda en el atributo que es un Resultset, y como este es público, las clases inferiores pueden acceder a él fácilmente, y así manipular la información.

```
/*************************************************************
 * Ejecutará las sentencia SQL de selección de información
 *************************************************************/
public void ObtenerInformacion(String strSentencia)
{       try
        {       stmt = con.createStatement();
                rsInformacion = stmt.executeQuery(strSentencia);
        }
        catch(SQLException ex)
        {
                objMensaje_err.EscribeMensaje_Err("TRANSACCION.ObtenerInforma
                cion(String strSentencia)");
        }
}
```

Lo mismo pasa con el método ActualizaInformacion(), la diferencia es que este método recibe sentencias del tipo UPDATE, INSERT o DELETE, y el resultado se guarda en el atributo intRegistrosAfectados miembro de la clase TRANSACCIÓN, este al igual que el Resultset es público, la diferencia es que se invoca al método executeUpdate() del objeto stmt, que se encarga de ejecutar precisamente este tipo de sentencias.

```
/*****************************************************************
 * Ejecutará las sentencia SQL de Inserción y actualización de información
 *****************************************************************/
public void ActualizaInformacion(String strSentencia)
{       try
        {       stmt = con.createStatement();
                System.out.println("Obteniendo Información.....un momento por favor");
                intRegistrosAfectados = this.stmt.executeUpdate(strSentencia);
                bolExiste_Error_TRANSACCION = false;
        }
        catch(SQLException ex)
        {       objMensaje_err.EscribeMensaje_Err("TRANSACCION.ActualizaInformacion(String strSentencia)
                *  + ex.getMessage());
                bolExiste_Error_TRANSACCION = true;
                System.out.println("Ha ocurrido un error en TRANSACCION.ActualizaInformacion()"
                                                        + "SQLException: " + ex.getMessage());      }
}
```

El método que sigue de la clase TRANSACCIÓN es Inserta(StringBuffer,StringBuffer,strTabla), este método recibe como parámetros dos StringBuffer strBValores que son los valores del registro a insertar, strBTipoValores que me indica qué tipo de valores contiene el StringBuffer anterior_ indicando 0 – si el valor es cadena y 1 – si el valor es numérico, esto es para decidir si el valor se cubre con apóstrofes o no, String strTabla contiene el nombre de la tabla a la cual se le va insertar el registro. Por lo tanto este método contiene la rutina que ensamblará cualquier sentencia INSERT.

```
/*****************************************************************
 * Arma y manda a ejecutar la instrucción INSERT
 *****************************************************************/
protected void Inserta(StringBuffer strBValores, StringBuffer strBTipoValores, String strTabla)
{       StringBuffer strBSentencia = new StringBuffer();
        StringBuffer strBTemporal = new StringBuffer();

        StringTokenizer strTkValores =  new StringTokenizer(strBValores.toString(),",");
        StringTokenizer strTkTipoValores =  new StringTokenizer(strBTipoValores.toString(),",");
        int intRegistros = strTkValores.countTokens();
        int intRegActual = 0;
        String strMensaje = new String();
        try
        {       strBSentencia.append("INSERT INTO " + strTabla);
                strBSentencia.append(" VALUES( ");
                intRegActual = 0;
                while(strTkValores.hasMoreTokens())
                {       intRegActual ++;
                        String strTipo = strTkTipoValores.nextToken();
                        System.out.println("Tipo Campos SETCriterios: " + strTipo);

                        if(strTipo.compareTo("0") == 0)
                        {       strBSentencia.append("'" + strTkValores.nextToken() + "'");
                                if(intRegActual != intRegistros)
                                        strBSentencia.append(" , ");        }
                        else
                        {       strBSentencia.append(strTkValores.nextToken());
                                if(intRegActual != intRegistros)
                                        strBSentencia.append(" , ");        }
                }
                strBSentencia.append(" )");
                this.ActualizaInformacion(strBSentencia.toString());
        }
        catch(Exception e)
        {       objMensaje_err.EscribeMensaje_Err("Error: TRANSACCION.Inserta() " +e.getMessage() );
        }
}
```

Lo mismo pasa con el método Actualiza(), solo que aquí recibe como parámetros todos aquellos elementos necesarios para ensamblar una sentencia UPDATE, expliquémoslo así, en el siguiente esquema, represento con diferente tipo de letra a los elementos del UPDATE que representa cada parámetro:

- **strBValores**
- strBTipoValores
- **strBValoresCriterios**
- strBTipoValorescriterios
- ***strBCamposSET***
- strBCamposCriterios
- ***strTabla***

Sentencia de Ejemplo:
UPDATE ***RH_EMPLEADO***
SET ***STR_APPATERNO*** = 'CAMPUSANO', ***STR_LUGAR_NACIMIENTO*** = 'DISTRITO FEDERAL'
WHERE INT_COD_EMPLEADO = **14**
Y los que contienen tipo de valores, cumplen la misma función que los del método Inserta().

```
/**********************************************************
* Arma y manda a ejecutar la instrucción UPDATE
**********************************************************/
protected void Actualiza(StringBuffer strBValores, StringBuffer strBTipoValores, StringBuffer strBValoresCriterios,
                         StringBuffer strBTipoValoresCriterios, StringBuffer strBCamposSET,
                         StringBuffer CamposCriterios, String strTabla)
{
        StringBuffer strBSentencia = new StringBuffer();
        StringBuffer strBTemporal = new StringBuffer();

        StringTokenizer strTkValores = new StringTokenizer(strBValores.toString(),",");
        StringTokenizer strTkCamposSET = new StringTokenizer(strBCamposSET.toString(),",");
        StringTokenizer strTkTipoValores = new StringTokenizer(strBTipoValores.toString(),",");

        StringTokenizer strTkValoresCriterios = new StringTokenizer(strBValoresCriterios.toString(),",");
        StringTokenizer strTkCamposSETCriterios = new StringTokenizer(strBCamposCriterios.toString(),",");
        StringTokenizer strTkTipoValoresCriterios = new StringTokenizer(strBTipoValoresCriterios.toString(),",");

        int intRegistros = strTkCamposSET.countTokens();
        int intRegActual = 0;
        String strMensaje = new String();
        try
        {       strBSentencia.append("UPDATE " + strTabla);
                strBSentencia.append(" SET ");
                while(strTkCamposSET.hasMoreTokens())
                {       intRegActual ++;
                        strBSentencia.append(strTkCamposSET.nextToken() + " = ");
                        String strTipo = strTkTipoValores.nextToken();
                        if(strTipo.compareTo("0") == 0 )
                        {       String strValores = strTkValores.nextToken();
                                strBSentencia.append("'" + strValores + "'");
                                if(intRegActual != intRegistros)
                                        strBSentencia.append(", ");         }
                        else{   String strValores = strTkValores.nextToken();
                                strBSentencia.append(strValores);
                                if(intRegActual != intRegistros)
                                        strBSentencia.append(", ");         }
                }
                this.ActualizaInformacion(strBSentencia.toString());
        }catch(Exception e)
        {       objMensaje_err.EscribeMensaje_Err("Error: TRANSACCION.Actualiza() "          }
}
```

Ahora bien, veamos nuestra clase CINVENTARIO, que de acuerdo con el esquema básico de nuestra jerarquía de clases, vendría siendo la CLASE_MODULO_SISTEMA, hereda directamente de la clase TRANSACCION, CINVENTARIO contendrá la integración completa de las sentencias SQL, es decir, esta clase contendrá un método por cada SELECT específico que se requiera, en este caso, el servlet correspondiente realizará este requerimiento. Por lo tanto existen varios métodos, así que solo presentaré los principales, ya que en el caso de los métodos que contienen sentencias del tipo SELECT, lo único que cambia entre ellos es la sentencia. Recordemos nuestra clase en la Fig. 4.5 y el código fuente de la definición de esta clase CINVENTARIO en la Fig. 4.6 , posteriormente veremos uno de sus métodos.

Figura 4.5
Clase CINVENTARIO

```
/**********************************************************************
        Creación:  14 - Abril - 2003
        Última Actualización: 2 - Mayo - 2003
        Clase CMANTENIMIENTO
        Contiene los métodos que se utilizarán en el Mantenimiento de los
        diferentes módulos del          Sistema de Recursos Humanos (SRH)

                                Edgar Herrera Casas
                                Tesis:Análisis, Diseño y Desarrollo del Sistema de Recursos Humanos
                                para La Sección Amarilla
**********************************************************************/
import java.sql.*;
import java.awt.* ;
import java.io.*;
import java.text.*;
import java.util.*;
public class CINVENTARIO extends TRANSACCION
{
        public boolean bolExiste_Error_CATALOGO;
        private static MENSAJE_ERR objMensaje_err;
        /**********************************************************************
         * Constructor: Llama al constructor de su Super Clase
         **********************************************************************/
        public CINVENTARIO(MENSAJE_ERR objMensaje_err_Global)
        {
                super(objMensaje_err_Global);
                objMensaje_err = (MENSAJE_ERR)objMensaje_err_Global;
                bolExiste_Error_CATALOGO = false;

        }
        /**********************************************************************
        * Selecciona Registro del empleado elegido
        **********************************************************************/
        public void SeleccionaRegistroRFCEmpleado(StringBuffer strBValoresCriterios)
        {       String strSentencia = new String();
                strSentencia = "SELECT STR_REGISTRO"+ " FROM rh_registro"+ " WHERE SMI_COD_REGISTRO = 0"
                                                + " AND INT_COD_EMPLEADO = " + ValoresCriterios.toString();
                try
                {       System.out.println("SENTENCIA: " + strSentencia);
                        this.ObtenerInformacion (strSentencia);
                        if(bolExiste_Error_TRANSACCION = false)
                                objMensaje_err.BorraMensaje_Err();
                        else
                                throw new Exception();
                        bolExiste_Error_CATALOGO = false;
                }
                catch(Exception e)
                {       objMensaje_err.EscribeMensaje_Err("CINVENTARIO.SeleccionaRegistroRFCEmpleado() "
                                                        +e.getMessage());
                        bolExiste_Error_CATALOGO = true;
                }
        }
}
```

Figura 4.6
Código fuente de la Clase CINVENTARIO

Veamos el constructor de nuestra clase, al igual que las otras recibe como parámetro el objeto objMensaje_err, que a su vez lo pasa a su superclase e inicializamos bolExiste_Error_CATALOGO = false indicando que aún no existe error.

```
/**************************************************************
 * Constructor: Llama al constructor de su Super Clase
 **************************************************************/
public CINVENTARIO(MENSAJE_ERR objMensaje_err_Global)
{
        super(objMensaje_err_Global);
        objMensaje_err = (MENSAJE_ERR)objMensaje_err_Global;
        bolExiste_Error_CATALOGO = false;

}
```

Como se ha mencionado anteriormente, esta clase posee los métodos que ensamblan consultas SQL especificas, el siguiente método se encarga de ensamblar la sentencia SQL que obtiene el RFC del empleado especificado, y se manda a llamar al método ObtenerInformacion() de nuestra clase TRANSACCIÓN vista anteriormente, se pregunta si ha ocurrido un error en la clase TRANSACCIÓN, si es asi se genera una excepción para dar a conocer dicho error. Así como este método son todos los demás métodos de esta clase, solo cambia la sentencia SQL.

```
/**************************************************************
 * Selecciona Registro del empleado elegido
 **************************************************************/
public void SeleccionaRegistroRFCEmpleado(StringBuffer strBValoresCriterios)
{       String strSentencia = new String();
        strSentencia = "SELECT STR_REGISTRO"+ " FROM rh_registro"+ " WHERE SMI_COD_REGISTRO = 0"
                                        + " AND INT_COD_EMPLEADO = " + ValoresCriterios.toString();
        try
        {       System.out.println("SENTENCIA: " + strSentencia);
                this.ObtenerInformacion (strSentencia);
                if(bolExiste_Error_TRANSACCION = false)
                        objMensaje_err.BorraMensaje_Err();
                else
                        throw new Exception();
                bolExiste_Error_CATALOGO = false;
        }
        catch(Exception e)
        {       objMensaje_err.EscribeMensaje_Err("CINVENTARIO.SeleccionaRegistroRFCEmpleado() "
                                                +e.getMessage());
                bolExiste_Error_CATALOGO = true;
        }
}
```

Sigamos subiendo de nivel, y ahora veamos nuestro Servlet, en este caso llamado CinventarioServlet, recordemos la clase en la Fig 4.7 y veamos el código fuente en la Fig 4.8.

Figura 4.7
Clase CinvenatrioServlet

```
/**************************************************************************
        Creación:   14 - Abril - 2003
        Última Actualización:   1 - Mayo - 2003
        Servlet CInventarioServlet
        Se encarga de mandar a ejecutar las instrucciones de selección de
        información para todas las consultas realizadas por los diferentes módulos
        de SRH.
                                Edgar Herrera Casas
                                Tesis:Análisis, Diseño y Desarrollo del Sistema de Recursos Humanos
                                para La Sección Amarilla
**************************************************************************/
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;
public class CInventarioServlet extends HttpServlet
{       private CINVENTARIO objInventario;          //Objeto que se encarga de realizar el mantenimiento
        private static MENSAJE_ERR objMensaje_err;  //Objeto almacena dónde ocurrió algún error
        int intRegistro;
        /**************************************************************************
        * Inicio del Servlet
        **************************************************************************/
        public void init(ServletConfig config) throws ServletException
        {       super.init(config);
                objMensaje_err = new MENSAJE_ERR();
                objInventario = new CINVENTARIO(objMensaje_err);
                intRegistro = 0;
        }
        public void doGet(HttpServletRequest req,HttpServletResponse response) throws ServletException,IOException {

                StringBuffer strBMensajeServlet = new StringBuffer();
                Enumeration enumVector;
                String strMensaje = new String();
                StringBuffer strBRegistro;

                //open InputStream
                System.out.println("ObjectInputStream input=new");
                ObjectInputStream input=new ObjectInputStream(req.getInputStream());
                // open OutputStream
                System.out.println("ObjectOutputStream output=new");
                try {   System.out.println("Vector objVector = (Vector)input.readObject();");
                        Vector objVector = (Vector)input.readObject();
                        strMensaje = "Se guardará en los objetos String";
                        enumVector = objVector.elements();
                        System.out.println("Servlet: Se obtiene el contenido del vector");
                        Integer intInstruccionSQL = (Integer)enumVector.nextElement();
                        StringBuffer strBTabla = new StringBuffer((String)enumVector.nextElement());
                        StringBuffer strBValores = new StringBuffer((String)enumVector.nextElement());
                        StringBuffer strBTipoValores = new StringBuffer((String)enumVector.nextElement());
                        StringBuffer strBValoresCriterios = new StringBuffer((String)enumVector.nextElement());
                        StringBuffer strBTipoValoresCriterios = new StringBuffer((String)enumVector.nextElement());
                        StringBuffer strBCamposSET = new StringBuffer();
                        StringBuffer strBCamposWHERE = new StringBuffer();
```

```
                    intRegistro = 0;
                    objVector.removeAllElements();
                    System.out.println("switch(intInstruccionSQL.intValue())");
                    switch(intInstruccionSQL.intValue())
                    {
                    case 0:
                            objInventario.Selecciona_MAX_Empleado();
                            while(objInventario.rsInformacion.next()){
                                    strBRegistro = new StringBuffer(objInventario.rsInformacion.getString(1));
                                    objVector.add(intRegistro,strBRegistro.toString());
                                    strBRegistro.delete(0,strBRegistro.length());
                                    intRegistro ++;}
                            break;
                    case 1:
                            //Catálogo de Nomina
                            objInventario.SeleccionaCatalogoNomina();
                            while(objInventario.rsInformacion.next()){
                                    strBRegistro = new StringBuffer(objInventario.rsInformacion.getString(1) + ","

                             + objInventario.rsInformacion.getString(2) + ","

                             + objInventario.rsInformacion.getString(3));
                                    objVector.add(intRegistro,strBRegistro.toString());
                                    strBRegistro.delete(0,strBRegistro.length());
                                    intRegistro ++;}
                            break;
                    }

            //Especificar el tipo del contenido de la respuesta
            strMensaje = "Especificar el tipo del contenido de la respuesta";
            response.setContentType("application/x-java-serialized-object");

            //Abrir el flujo de salida para enviar objetos
            strMensaje = "Abrir el flujo de salida para enviar objetos";
            ObjectOutputStream objOuputSSalida = new ObjectOutputStream(response.getOutputStream());
            strMensaje = "objOuputSSalida.writeObject(objVector);";
            objOuputSSalida.writeObject(objVector);
            //Asegurar que envie el contenido forzando el vaciado del buffer
            objOuputSSalida.flush();
            objOuputSSalida.close();
            //limpiar Buffers
            objInventario.rsInformacion = null;
            objVector.removeAllElements();

            }
            catch(Exception e)
            {

                    System.out.println("Servlet Error: " + e.getMessage());
                    e.printStackTrace();

            }
    }

    public void doPost(HttpServletRequest req,HttpServletResponse res) throws ServletException,IOException
    {
            doGet(req,res);
    }

}
```

Figura 4.8
Código fuente de la Clase
CInventarioservlet

Ahora bien, de lado del servidor aquí es donde empieza todo, en cuanto se invoca el servlet se ejecuta el método init(), y aquí es donde se crea el objeto objMensaje_err y objInventario, de este último objeto se desata nuestra jerarquía de clases según el esquema presentado en el capítulo IV, y se inicializa un atributo intRegistro = 0.

```
/**********************************************************************
* Inicio del Servlet
**********************************************************************/
public void init(ServletConfig config) throws ServletException
{       super.init(config);
        objMensaje_err = new MENSAJE_ERR();
        objInventario = new CINVENTARIO(objMensaje_err);
        intRegistro = 0;
}
```

Como el servlet es llamado por el método DoPost, después de init() se ejecuta este, que no hace mas que llamar al método doGet(), enviándole como parámetros los dos objetos del servlet HttpServletRequest y HttpServletResponse

```
public void doPost(HttpServletRequest req,HttpServletResponse res) throws ServletException,IOException
        {
                doGet(req,res);
        }
```

El método doGet(), es el que se encargará de canalizar las peticiones del applet hacia el objeto objtInventario, por medio de un switch. Para esto se lee la información encapsulada contenida en el objeto req del servlet, posteriormente se utiliza como contenedor de objetos un objeto Vector, posteriormente se obtiene la información almacenando en StringBuffer's y un Integer que es el que indicará específicamente cuál es el requerimiento del applet.

```
ObjectInputStream input=new ObjectInputStream(req.getInputStream());
        // open OutputStream
        System.out.println("ObjectOutputStream output=new");
        try {   System.out.println("Vector objVector = (Vector)input.readObject();");
                Vector objVector = (Vector)input.readObject();
                strMensaje = "Se guardará en los objetos String";
                enumVector = objVector.elements();
                System.out.println("Servlet: Se obtiene el contenido del vector");
                Integer intInstruccionSQL = (Integer)enumVector.nextElement();
                StringBuffer strBTabla = new StringBuffer((String)enumVector.nextElement());
                StringBuffer strBValores = new StringBuffer((String)enumVector.nextElement());
                StringBuffer strBTipoValores = new StringBuffer((String)enumVector.nextElement());
                StringBuffer strBValoresCriterios = new StringBuffer((String)enumVector.nextElement());
                StringBuffer strBTipoValoresCriterios = new StringBuffer((String)enumVector.nextElement());
                StringBuffer strBCamposSET = new StringBuffer();
                StringBuffer strBCamposWHERE = new StringBuffer();
                intRegistro = 0;
                objVector.removeAllElements();
                System.out.println("switch(intInstruccionSQL.intValue())");
                switch(intInstruccionSQL.intValue())
                {
                case 0:
                        objInventario.Selecciona_MAX_Empleado();
                        while(objInventario.rsInformacion.next()){
                                strBRegistro = new StringBuffer(objInventario.rsInformacion.getString(1));
                                objVector.add(intRegistro,strBRegistro.toString());
                                strBRegistro.delete(0,strBRegistro.length());
                                intRegistro ++;}
                        break;
```

Posteriormente se crea un objeto de salida, para enviar mediante este el objeto Vector, que contiene el resultado de la consulta en turno, se limpia el flujo de salida; y se cierra, así como limpiar el objeto Vector.

```
//Especificar el tipo del contenido de la respuesta
response.setContentType("application/x-java-serialized-object");

//Abrir el flujo de salida para enviar objetos
ObjectOutputStream objOuputSSalida = new ObjectOutputStream(response.getOutputStream());
objOuputSSalida.writeObject(objVector);

//Asegurar que envie el contenido forzando el vaciado del buffer
objOuputSSalida.flush();
objOuputSSalida.close();
//limpiar Buffers
objInventario.rsInformacion = null;
objVector.removeAllElements();
```

La siguiente clase que presentaré es CInventarioApplet, en esta clase se alberga la Interfaz Gráfica de Usuario, y las rutinas que se comunican con ella. Presentaré los métodos mas importantes de esta clase, no sin antes recordarla en la Fig. 4.9, veamos su código fuente en el cual se define la clase en la Fig. 4.10.

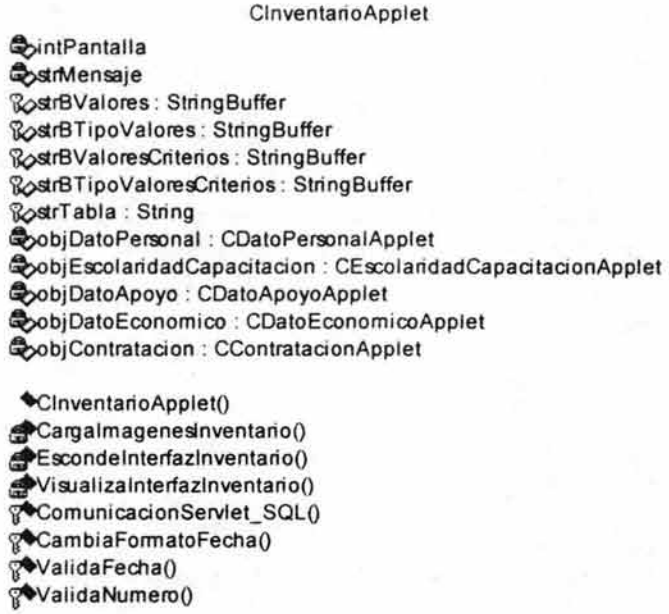


Figura 4.9
Clase CInventarioApplet

```
/**************************************************************************
        Creación:  22 - Marzo - 2003
        Última modificación:  24 - Julio - 2003
        Applet CInventarioApplet
        Se encargará de desplegar en el browser la Pantalla Inventario de Recursos
        Humanos del Sistema de Recursos Humanos (SRH)
                        Edgar Herrera Casas
                        Tesis:Análisis, Diseño y Desarrollo del Sistema de Recursos Humanos
                        para La Sección Amarilla
**************************************************************************/
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
import java.net.*;
import java.io.*;
import java.util.*;
import java.sql.*;
import javax.swing.*;
import javax.swing.event.*;
import javax.swing.table.*;
public class CInventarioApplet extends CSRHApplet
{
        //Objetos para las pantallas de este Módulo
        CDatoPersonalApplet objDatoPersonal;
        CEscolaridadCapacitacionApplet objEscolaridadCapacitacion;
        CDatoApoyoApplet objDatoApoyo;
        CDatoEconomicoApplet objDatoEconomico;
        CContratacionApplet objContratacion;

        //Objetos y variables para la manipulación de información
        protected StringBuffer strBValores;
        protected StringBuffer strBTipoValores;
        protected StringBuffer strBValoresCriterios;
        protected StringBuffer strBTipoValoresCriterios;
        protected String strTabla;
        /*********************************************************
        *       Se encarga de informar en qué pantalla se encuentra
        *       actualmente el usuario
        *       0 - Datos Personales
        *       1 - Escolaridad y Capacitación
        *       2 - Datos de Apoyo
        *       3 - Datos Económicos
        *       4 - Contratación
        *********************************************************/
        private int intPantalla;
        private String strMensaje;
        /*************************************************************************
        * Se encarga de crear los objetos en los cuales se alberagará toda la interfaz del módulo
        *************************************************************************/
        public CInventarioApplet(Container contentPane_Global,JPanel panelButtons_Global)
        {       panelInventario = new JPanel();
                contentPane = getContentPane();
                try{    contentPane = (Container)contentPane_Global;
                        panelInventario = (JPanel)panelButtons_Global;
                        intPantalla = -1;
                        panelInventario.setLayout(null);
                        contentPane.setLayout(null);
                        CargaImagenesInventario();
                        InterfazInventario();           }
                catch(Exception e){}
        }
}
```

Figura 4.10
Código fuente de la Clase CInventarioApplet

Esta clase es muy importante ya que de aquí partirán todas las clases de este módulo. Hereda de la clase CSRHApplet, y heredan de ella cinco clases del módulo, por lo tanto estas clases heredan los atributos:

```
protected StringBuffer strBValores;
protected StringBuffer strBTipoValores;
protected StringBuffer strBValoresCriterios;
protected StringBuffer strBTipoValoresCriterios;
protected String strTabla;
```

Que tienen la función de almacenar en el momento que se requiera, toda aquella información para actualizar nuestra base de datos, estos atributos son los que se utilizan como parámetros para enviárselos al servlet visto anteriormente mediante un objeto Vector. El atributo intPantalla se explica en el mismo código fuente con comentarios. En el constructor se reciben como parámetros, un Container y un JPanel, estos no son mas que contenedores de objetos interfaz de los componentes swing, estos provienen de nuestra clase principal CSRHApplet, posteriormente se manda a llamar dos métodos de esta misma clase; CargaImagenesInventario() y InterfazInventario(), que se encargan de cargar todos lo íconos que serán utilizados en la interfaz de esta clase y crear la misma correspondientemente.

```
/**************************************************************************
* Se encarga de crear los objetos en los cuales se alberagará toda la interfaz del módulo
**************************************************************************/
public CInventarioApplet(Container contentPane_Global,JPanel panelButtons_Global)
{       panelInventario = new JPanel();
        contentPane = getContentPane();
        try{    contentPane = (Container)contentPane_Global;
                panelInventario = (JPanel)panelButtons_Global;
                intPantalla = -1;
                panelInventario.setLayout(null);
                contentPane.setLayout(null);
                CargaImagenesInventario();
                InterfazInventario();           }
        catch(Exception e){}
}
```

Ahora presentaré un método muy importante, que es el encargado de comunicarse con el servlet CinventarioServlet:, llamado ComunicacionServlet_SQL(int), este recibe como único parámetro un entero que es el mensaje que le va a indicar al servlet qué mensaje este enviará al objeto objInventario para que se ejecute la sentencia SQL correspondiente. Se utiliza un Vector para almacenar toda la información necesaria para el servlet, se crea un objeto URL que contendrá la dirección del servlet, posteriormente se crea un objeto URLConnection que es el que me servirá para abrir la conexión al servlet, configurar esta y crear un objeto ObjectOutputStream que servirá como flujo de salida para enviar toda la información recabada en un Vector, se vacía el buffer de salida y se cierra dicho flujo, del lado del servidor se ejecuta todo lo ya visto anteriormente a partir del servlet, y este regresa también en un Vector la información obtenida desde el servidor, en el applet como todavía tenemos la conexión abierta, la utilizamos para que por medio de un objeto ObjectInputStream se lea lo que nos regresa el servlet y se guarda en el Vector ya creado, este es el Vector que a su vez regresa nuestro método al método de dónde fue invocado, para que la información sea manipulada.

```
/**********************************************************************
* Se encarga de Comunicarse con el servlet correspondiente para seleccionar
* la información especificada
* el único parametro indica el tipo de SELECT que se realizará, es decir
* con este número se indica al servlet qué SELECT ejecutará o bien:
* 6 - CUALQUIER INSERT
**********************************************************************/
protected Vector ComunicacionServlet_SQL(int intInstruccionSQL)
{       Vector objVector = new Vector();
        StringBuffer strBMensaje = new StringBuffer();
        String strBaseURL = "http://localhost:8080/servlet/CInventarioServlet";
        try
        {       objVector.add(0,new Integer(intInstruccionSQL));
                objVector.add(1,strTabla.toString());
                objVector.add(2,strBValores.toString());
                objVector.add(3,strBTipoValores.toString());
                objVector.add(4,strBValoresCriterios.toString());
                objVector.add(5,strBTipoValoresCriterios.toString());

                URL direccion = new URL(strBaseURL);
                //Abrir la conexión al Servlet
                URLConnection servletConnection = direccion.openConnection();

                //Decirle al navegador que no use su caché para esta conexión,
                //porque si lo hace, tendremos una página estática
                servletConnection.setUseCaches(false);

                //La conexión es informada que aceptará datos de retorno del servlet
                servletConnection.setDoInput(true);
                //La conexión es informada que enviará información al Servlet
                servletConnection.setDoOutput(true);

                //Añadir las cabeceras de HTTP
                servletConnection.setRequestProperty("Content-Type","application/octet-stream");

                //Leer el ObjectOutputStream del Servlet
                ObjectOutputStream objOuputSSalida = new ObjectOutputStream(

                servletConnection.getOutputStream());
                //Mandar los objetos al servlet por el flujo de salida
                objOuputSSalida.writeObject(objVector);
                //Asegurar que envie el contenido forzando el vaciado del buffer
                objOuputSSalida.flush();
                objOuputSSalida.close();

                //Leer el ObjectInputStream desde el Servlet
                ObjectInputStream ObjInputSEntrada = new ObjectInputStream(

                servletConnection.getInputStream());
                objVector = (Vector)ObjInputSEntrada.readObject();
        }
        catch(Exception e)
        {       JOptionPane.showMessageDialog(btnSRHPrincipal," Error","SRH - Inventario de
                Recursos Humanos",JOptionPane.INFORMATION_MESSAGE);     }
        return objVector;
}
```

Ahora echemos un vistazo a la clase CSRHApplet, esta es llamada desde una JSP o HTML, recordémosla en la Fig. 4.11 y su código fuente en la Fig. 4.12.

Figura 4.11
Clase CSRHApplet

```
/**********************************************************************
        Creación:  27 - Noviembre - 2002
        Última modificación:  2 - Mayo - 2003
        Applet CSRHApplet
        Se encargará de desplegar en el browser la Pantalla Principal del
        Sistema de Recursos Humanos (SRH)

                        Edgar Herrera Casas
                        Tesis: Análisis, Diseño y Desarrollo del Sistema de Recursos Humanos
                        para La Sección Amarilla
**********************************************************************/
import java.applet.*;
import java.awt.*;
import java.awt.Color;
import java.awt.event.*;
import java.net.*;
import java.io.*;
import java.util.*;
import java.sql.*;
import javax.swing.*;
import javax.swing.event.*;
import javax.swing.table.*;
import com.sun.java.swing.plaf.motif.MotifLookAndFeel;
import javax.swing.plaf.metal.MetalLookAndFeel;
import javax.swing.plaf.ColorUIResource;
public class CSRHApplet extends JApplet
{       URL direccion;
        ImageIcon icon;
        Container content;
        JPanel panelSRH;
        CInventarioApplet objInventario;
        CNominaApplet objNomina;
        CCapacitacionApplet objCapacitacion;
        CReclutamientoSeleccionApplet objReclutamientoSeleccion;
        CAdministracionSistemaApplet objAdministracionSistema;
        CIMSSApplet objIMSS;
        String strMensaje;
        public void init()
        {       InterfazPrincipal();
        }
}
```

Figura 4.12
Código fuente de la Clase CSRHApplet

Como podemos apreciar en nuestro código de la clase CSRHApplet, esta hereda de la clase JApplet, esto quiere decir que es un applet y todas las clases que hereden de ella también lo serán, en la definición de nuestra clase no hacemos mas que declarar nuestros objetos de lo que van a ser nuestros módulos principales del Sistema, un atributo mas strMensaje que nos servirá para visualizar los posibles errores, el método init() característico de una applet, manda a llamar a nuestro método local InterfazPrincipal(), que se encarga de mostrar lo que serán los accesos a nuestros módulos, estos están hechos con componentes swing de java.

Nuestra clase CSRHApplet para que sea visualizada en un browser, es llamada desde una JSP, aquí es de donde en realidad parte la ejecución de nuestro sistema, desde un JSP, en este se especifica el nombre del applet principal, es decir, su archivo CLASS, se indica el tamaño que ocupará dicha applet y la versión del compilador que estamos utilizando. Veamos el código de la Fig. 4.13.

```
<%--
Creación:01 - Diciembre - 2002
        Última modificación:        11 - Enero - 2003
        JSP CSRHApplet.jsp
        Se encarga de llamar al applet principal del Sistema
                        Edgar Herrera Casas
                        Tesis:Anàlisis, Diseño y Desarrollo del Sistema de Recursos Humanos
                        para La Sección Amarilla
--%>
<HTML>
<BODY bgColor=#cccccc>
</BODY>


<%@page session="false" %>
<jsp:plugin
        type="applet"
        code="CSRHApplet.class"
        codebase="."
        width="1130"
        height="725 "
        jreversion="1.3"
        >
</jsp:plugin>
</HTML>
```

Figura 4.13
Código fuente de la JSP SRH.jsp

Hemos presentado el desarrollo de nuestro primer módulo y nuestra clase principal del sistema, ahora bien, **la metodología que hemos visto, es la misma que se utilizará para el resto del sistema, es decir, cada módulo principal contará con un applet principal, su servlet correspondiente, así como todos los demás submódulos, como quien dice, se repetirá la historia en cada uno de estos.**

4.3. Desarrollo de la Interfaz Gráfica de Usuario

4.3.1. Componentes Swing

```
import javax.swing.*;
import javax.swing.event.*;
import javax.swing.table.*;
import javax.swing.border.*;
```

Estos son algunos de los paquetes que se utilizaron para desarrollar la interfaz gráfica del sistema. Veamos en el siguiente recuadro el método interfaz de nuestra superclase del sistema, este se encarga de construir la primer ventana del mismo, la de seguridad, como podemos observar, no es mas que crear objetos de las clases definidas en los paquetes anteriormente expuestos.

```
private void Interfaz()
{
        lblUsuario = new JLabel("Nombre se Usuario");
        lblContraseña = new JLabel("Contraseña");
        txtUsuario = new JTextField();
        txtContraseña = new JTextField();

        btnAceptar = new JButton("Aceptar");
        btnCancelar = new JButton("Cancelar");

        lblUsuario.setBounds(60,10,200,25);
        lblUsuario.setFont(new Font("Arial Narrow",Font.PLAIN,13));
        lblContraseña.setBounds(60,70,200,25);
        lblContraseña.setFont(new Font("Arial Narrow",Font.PLAIN,13));

        txtUsuario.setBounds(60,35,140,20);
        txtUsuario.setFont(new Font("Arial Narrow",Font.PLAIN,13));
        txtContraseña.setBounds(60,95,140,20);
        txtContraseña.setFont(new Font("Arial Narrow",Font.PLAIN,13));

        btnAceptar.setBounds(50,135,80,25);
        btnAceptar.setFont(new Font("Arial Narrow",Font.PLAIN,13));
        btnAceptar.setToolTipText("Entrar al Sistema");
        btnAceptar.setMnemonic('A');
        btnAceptar.setHorizontalAlignment(JButton.CENTER);

        btnCancelar.setBounds(150,135,80,25);
        btnCancelar.setFont(new Font("Arial Narrow",Font.PLAIN,13));
        btnCancelar.setToolTipText("Salir");
        btnCancelar.setMnemonic('C');
        btnCancelar.setHorizontalAlignment(JButton.CENTER);

        panelSeguridad.add(lblUsuario);
        panelSeguridad.add(lblContraseña);
        panelSeguridad.add(txtUsuario);
        panelSeguridad.add(txtContraseña);
        panelSeguridad.add(btnAceptar);
        panelSeguridad.add(btnCancelar);

        dialogContentPaneSeguridad.add(panelSeguridad);
```

Continuiando con el método interfaz, este también contiene los eventos de cada objeto, en este caso los de los botones, estos objetos miembros de la clase Jbutton.

```
btnAceptar.addActionListener(new ActionListener()
{
        public void actionPerformed(ActionEvent e)
        {
                try
                {
                        CargarInformacionUsuario();
                        objCompañiaApplet = new CCompañiaApplet();

                        objCompañiaApplet.init(contentPane,objVectorInformacionUsuario);

                        dialogSeguridad.dispose();
                        repaint();
                }
                catch(Exception ex)
                {
                        System.out.println("Error: " + ex.getMessage());
                        showStatus("Error: " + ex.getMessage());
                }

        }
        });
        btnCancelar.addActionListener(new ActionListener()
        {
                public void actionPerformed(ActionEvent e)
                {
                        try
                        {
                                dialogSeguridad.dispose();
                        }
                        catch(Exception ex)
                        {
                                System.out.println("Error: " + ex.getMessage());
                                showStatus("Error: " + ex.getMessage());
                        }

                        }
                });
        }
```

Ahora veamos una pequeña explicación de los objetos interfaz que utiliza el método Interfaz:

- Jlabel. Los objetos miembros de esta clase son etiquetas prefijo lbl.
- JtextField. Los objetos miembros de esta clase son cajas de texto, son entradas del sistema prefijo txt.
- Jbutton. Los objetos miembros de esta clase son botones de comando prefijo btn.
- JDialog. Los objetos miembros de esta clase son ventanas de dialogo prefijo dialog.

Para activar los eventos de cada objeto implementamos un escuchador al objeto correspondiente:

```
btnCancelar.addActionListener(new ActionListener()
{
```

El método Interfaz se encuentra en cada módulo del sistema, así que no hacemos mas que crear y colocar los objetos de nuestra interfaz en cada ventana.

4.4. Documentación del Sistema

4.4.1. Manual de usuario de SRH

4.4.1.1. Entrar al Sistema

Para entrar al sistema, abrir el explorador de Windows o Netscape, en la barra de Dirección escribir la siguiente:

http://192.168.0.1/examples/jsp/srh10/SRH/SRH.html

Aparecerá la siguiente pantalla (Ver Fig:4.12.), dónde el usuario se firmará con su nombre de usuario y contraseña previamente proporcionadas por el administrador del sistema y dar un clic en el botón de Aceptar.

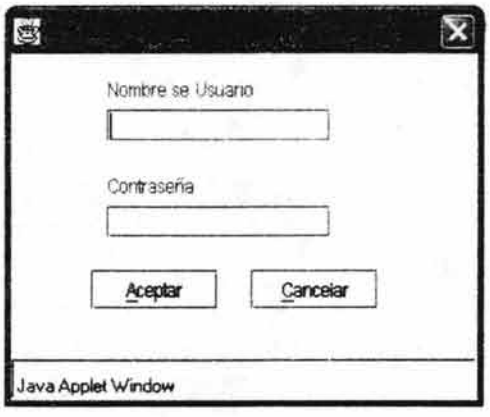


Figura 4.12
Pantalla de Seguridad del sistema

Una vez que el usuario capturó su nombre de usuario y contraseña, si estos son correctos, el sistema mostrará enseguida una pantalla dónde podrá elegir la compañía en la cual se va a trabajar, y dar clic en el botón de Aceptar (Ver Fig 4.13).

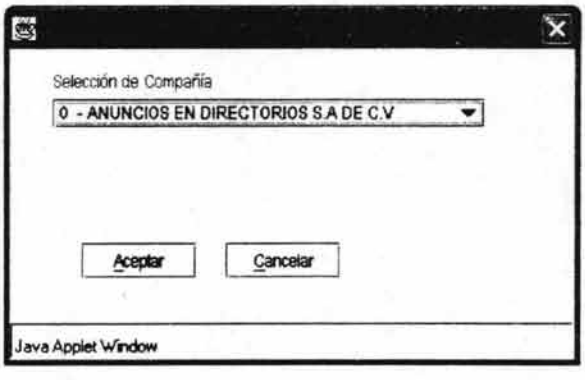


Figura 4.13
Pantalla de elección de compañía

Una vez que se eligió la compañía en la que se va a trabajar, el sistema mostrará la pantalla principal, dónde se pueden ver todos los módulos principales, estos estarán habilitados o deshabilitados de acuerdo al perfil de seguridad que posee el usuario, en la pantalla nos damos cuenta que el usuario que ingresó tiene perfil de administrador, ya que ninguno de los módulos está deshabilitado(Ver Fig 4.14).

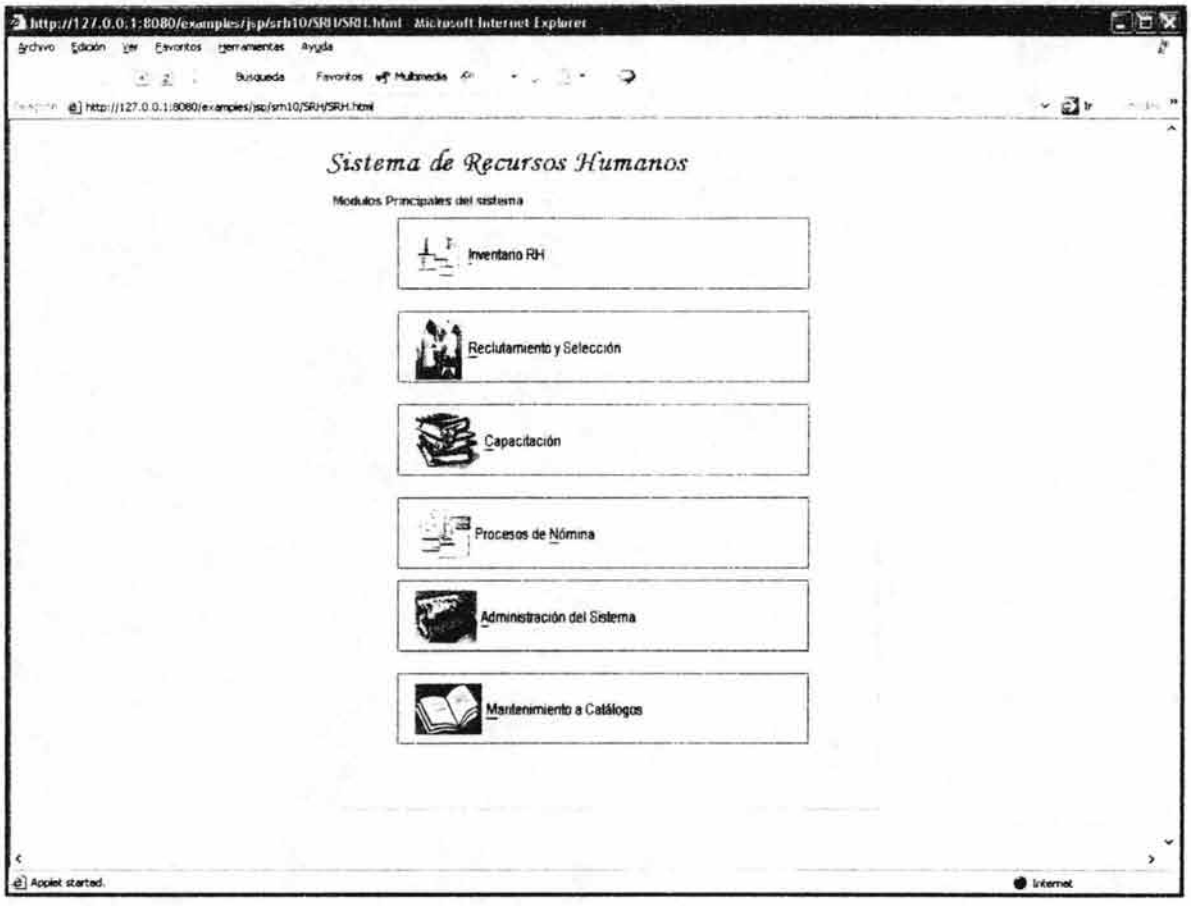


Figura 4.14
Pantalla Principal

4.4.1.2. Inventario de Recursos Humanos

Para dar mantenimiento al Inventario de Recursos Humanos, el procedimiento que se explicará a continuación, es el mismo para cada pantalla de este módulo. En el menú <Inventario de Recursos Humanos> damos un clic y nos mostrará todos los apartados disponibles, vamos a elegir la opción de menú <Datos Personales>, nos muestra la pantalla de este apartado.

4.4.1.2.1 Consultar Información del Empleado

Capturar el número de empleado[41] en el campo correspondiente, y dar clic en el botón <Cargar Empleado>, el sistema cargará la información del empleado, una vez cargada la información podrá se modificada, los cambios se guardan oprimiendo el botón <Actualizar>. Ver Fig. 4.15. Este mismo procedimiento es para los módulo de Capacitación y Reclutamiento y Selección.

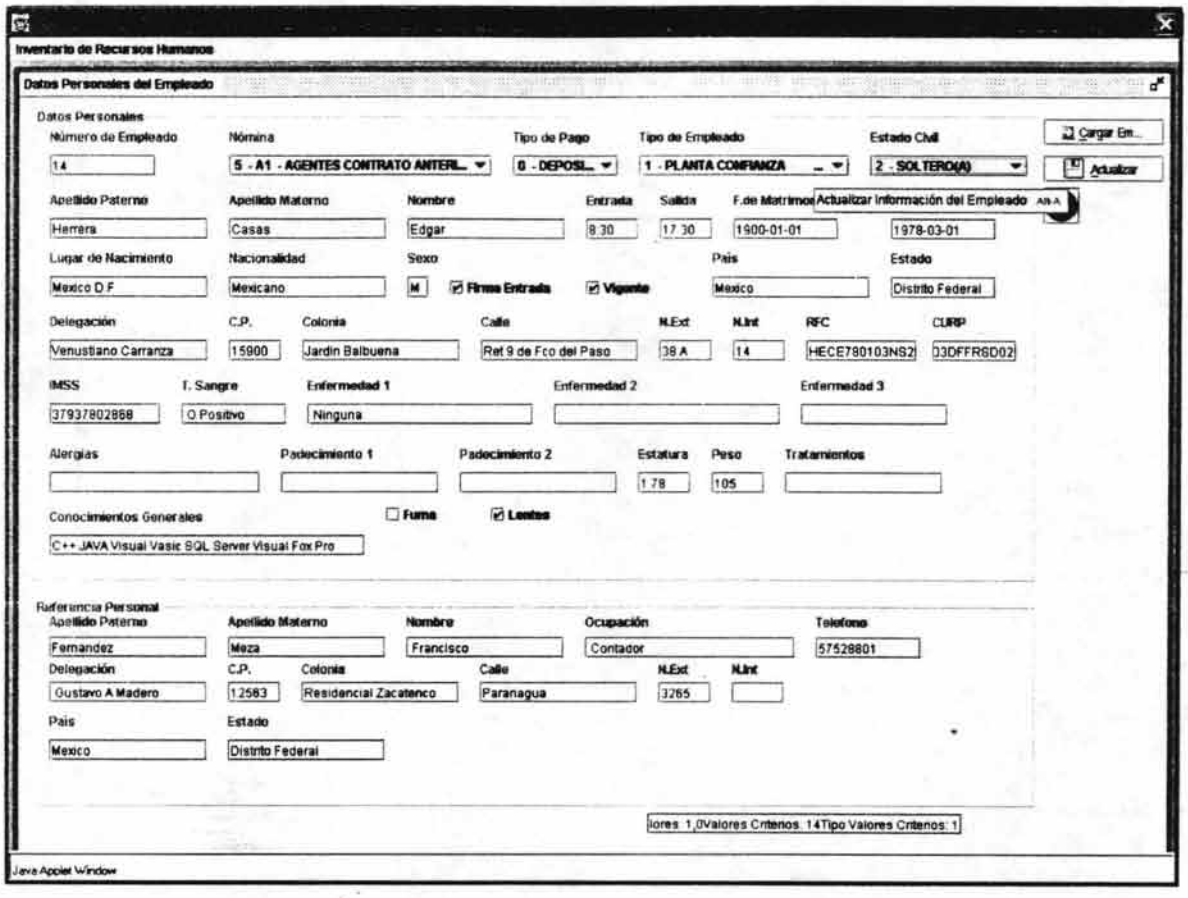


Figura 4.15
Pantalla de Mantenimiento a los datos Personales del empleado.

[41] El número de empleado es estrictamente numérico, por lo tanto, si se introduce un dato no numérico causará un error

4.4.1.2.2 Dar de alta al empleado

En la pantalla de Inventario de Recursos Humanos, en la opción de menú <Dar de Alta Empleado>, nos mostrará la primer pantalla para dar de alta al empleado, esta es la de Datos Personales, podemos apreciar que entes momento solo tenemos el botón de guardar, oprimiendo este si no existe ningún problema en la captura, guardará la información en la Base de Datos. El número de Empleado lo genera automáticamente el sistema, y no podrá se modificado por el usuario en ningún momento. Una vez que se halla acabado de capturar esta pantalla, deberá de respetar el orden de captura de acuerdo al las ventanas correspondientes y en el orden que se encuentran en el menú, es decir, una vez capturados los datos personales, se dispondrá a abrir la ventana de Escolaridad y Capacitación y guardar la información capturada, y así sucesivamente, solo así logrará capturar la información completa del empleado. Ver Fig. 4.15.

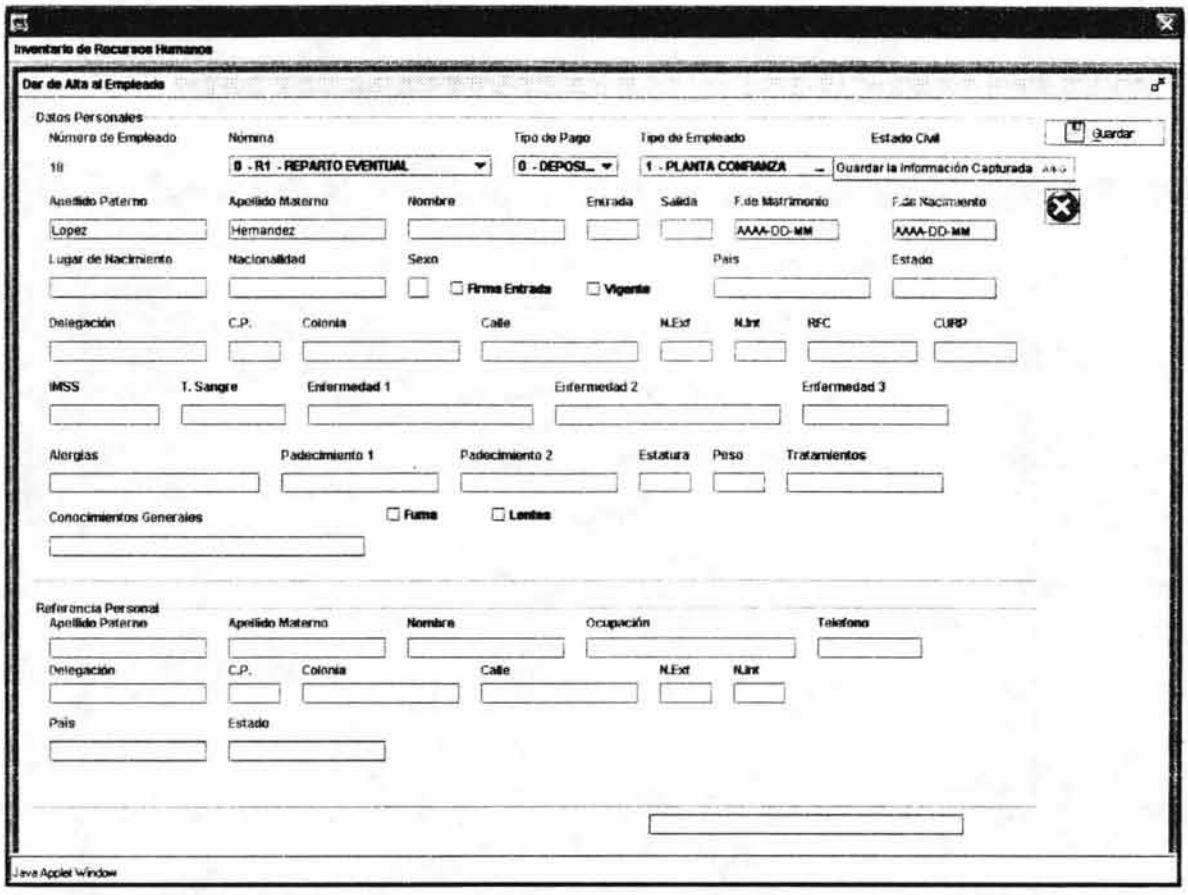


Figura 4.15
Primer Pantalla para dar de Alta al Empleado

4.4.1.3. Procesos de Nómina

4.4.1.3.1 Mantenimiento a Tablas de Nómina

4.4.1.3.1.1 Mantenimiento a las tablas de Impuesto

Encontrándonos en la ventana principal, oprimir el botón <Procesos de Nómina>, el sistema nos mostrará la pantalla del módulo, y en el menú <Procesos de Nómina>, seleccionar la opción <Tablas de Nómina>, veremos una pantalla como la de la Fig. 4.16, oprimir el botón <Tablas de Impuesto> y veremos una pantalla como la de la Fig. 4.17. Dentro de esta última en el menú <Tablas de Impuesto> seleccionar la opción <Capturar Nueva Tabla>, en el marco <Tabla Actual>, seleccionar qué tipo de tabla[42] de va a capturar, posteriormente capturar las fechas de inicio y término en el formato requerido; AAAA-DD-MM, incluyendo los guiones, ahora seleccionar del menú, la opción <Nuevo Registro>, nos aparecerá una fila en blanco, la cual seleccionamos con el mouse (en la primer columna), y seleccionamos la opción del menú <Modificart Registro seleccionado> y en los campos inferiores capturar el registro, Aceptar la acción con el botón que contiene una palomita. El registro se agregará a la tabla, y así sucesivamente con todos los registros, al final guardar la información seleccionado en el menú la opción de <Guardar Cambios>.

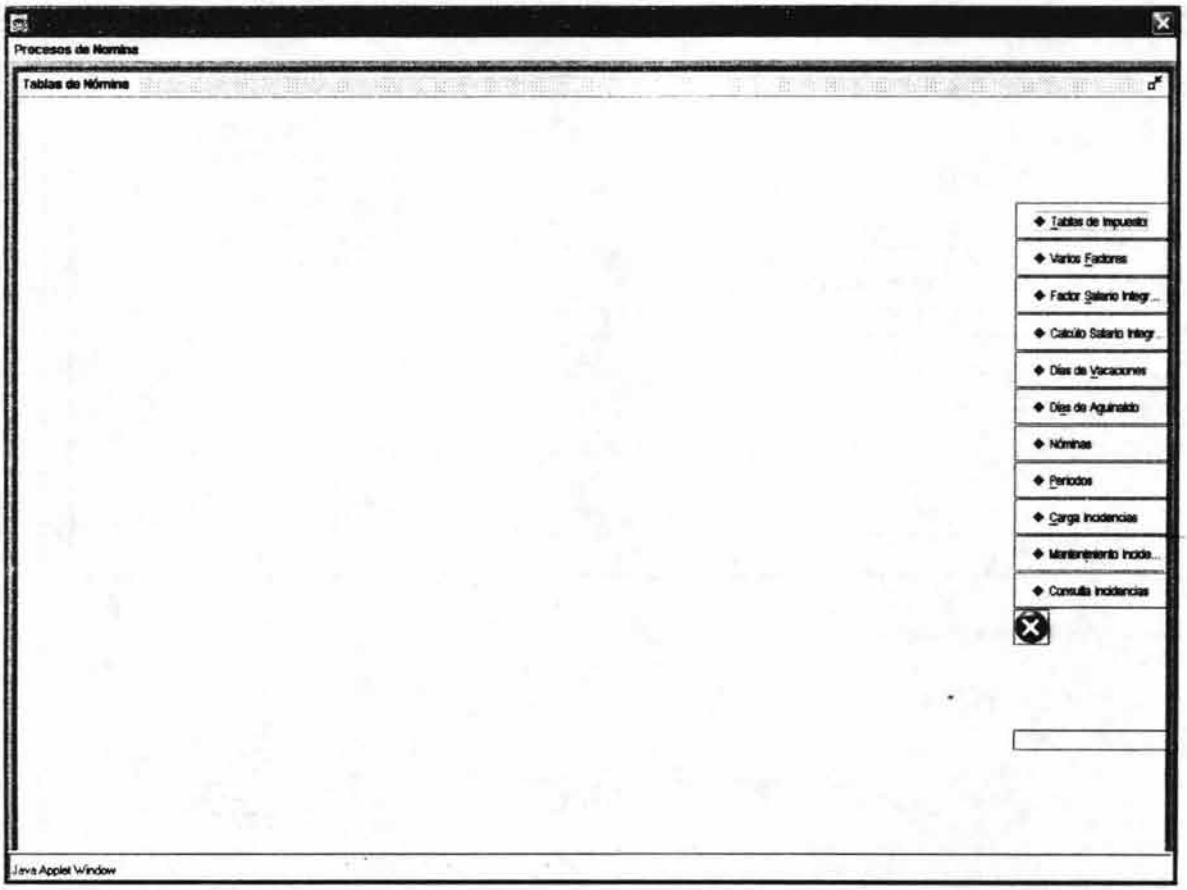


Figura 4.16
Pantalla para dar mantenimiento
a las tablas de Nómina

[42] Impuesto, Subsidio o Crédito al Salario

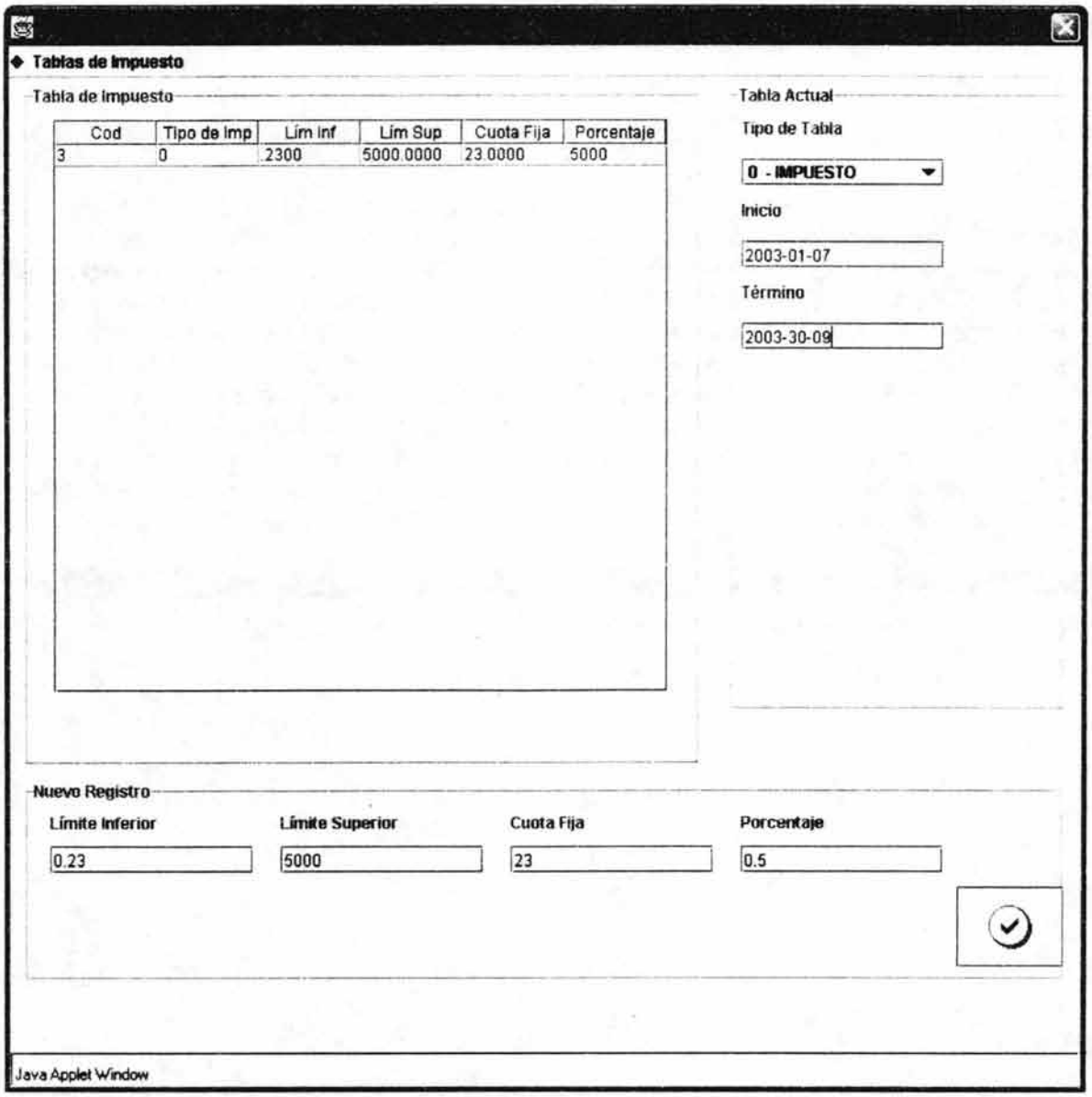


Figura 4.17
Pantalla para dar mantenimiento
a las tablas de Impuesto

4.4.1.3.1.2 Mantenimiento a varios factores

En este apartado solo se modifican los valores de los factores establecidos, seleccionado el factor, capturando su valor numérico y dar clic en <Guardar>. Ver Fig. 4.18

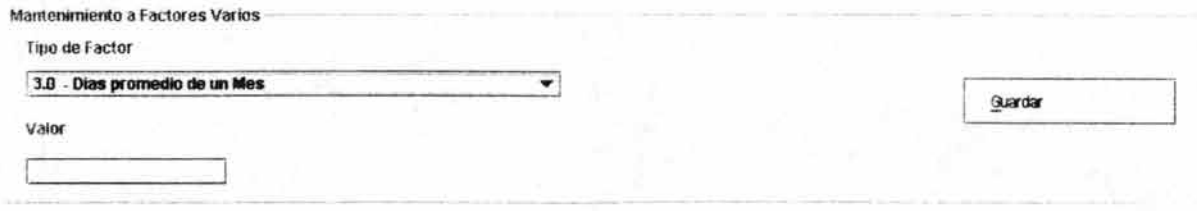


Figura 4.18
Pantalla para dar mantenimiento
a varios factores

4.4.1.3.1.2 Mantenimiento a Factores del Salario Integrado

Es el mismo procedimiento de "Mantenimiento a Tablas de Impuesto". Ver Fig. 4.19.

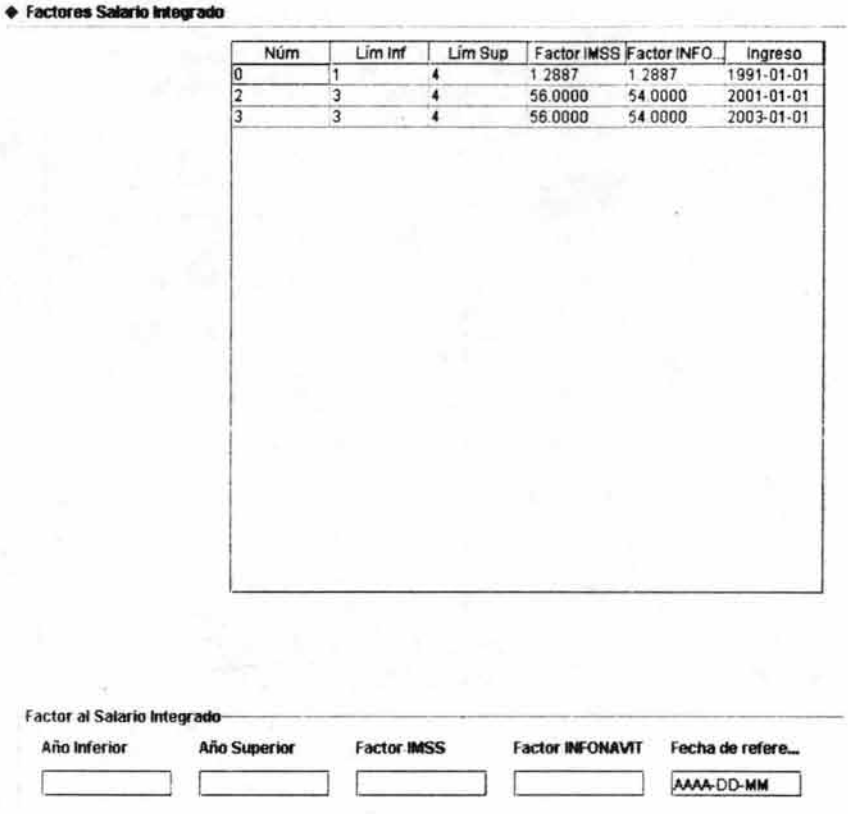


Figura 4.19
Pantalla para dar mantenimiento
a Factores del Salario Integrado.

4.4.1.3.1.4 Cálculo de Salarios Integrados

Se selecciona la nómina, el año y el intervalo de periodos que se va a tomar para el cálculo de Salarios Integrados, con el botón de <Calcular>, se realiza el cálculo, en este momento no se ha actualizado la Base de Datos, hasta que se haya dado clic al botón de <Actualizar>, antes se puede imprimir en un archivo el reporte de cómo quedarán los salarios integrados, esto con el botón de <Imprimir>.Ver Fig. 4.20.

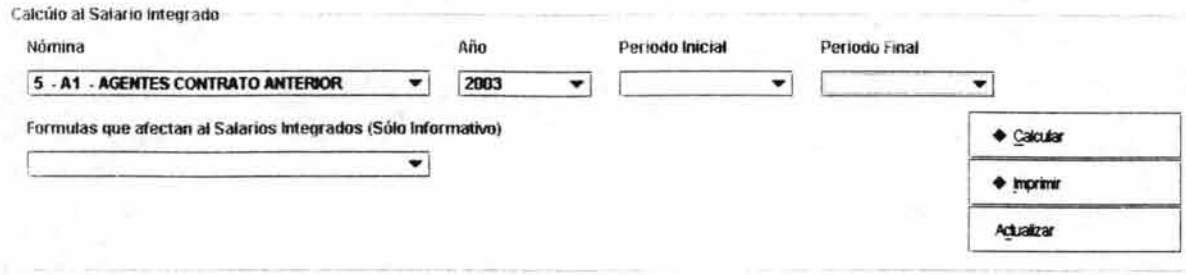


Figura 4.20
Pantalla para calcular Salarios Integrados.

4.4.1.3.1.5 Mantenimiento a Días de Vacaciones

Seleccione la Nómina, haga las modificaciones correspondientes, pudiendo eliminar registros con la opción de menú <Eliminar> o Agregar con la opción de menú <Insertar>, una vez realizados los cambios, con la opción de menú <Guardar Tabla> se actualizará la Base de Datos. Ver Fig. 4.21.

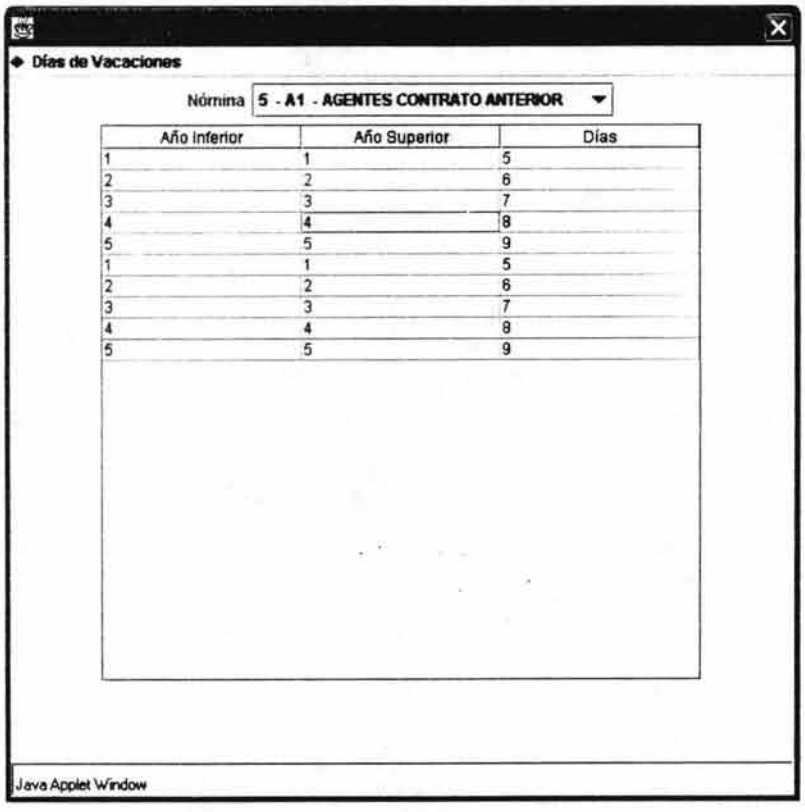


Figura 4.21
Mantenimiento a Días de Vacaciones

4.4.1.3.1.6 Mantenimiento a Días de Aguinaldo

Mismo procedimiento que "Mantenimiento a Varios Factores". Ver Fig. 4.22.

Figura 4.22
Mantenimiento a Días de Aguinaldo

4.4.1.3.1.7 Mantenimiento a Nóminas

Mismo procedimiento que "Mantenimiento a Varios Factores". Ver Fig. 4.23.

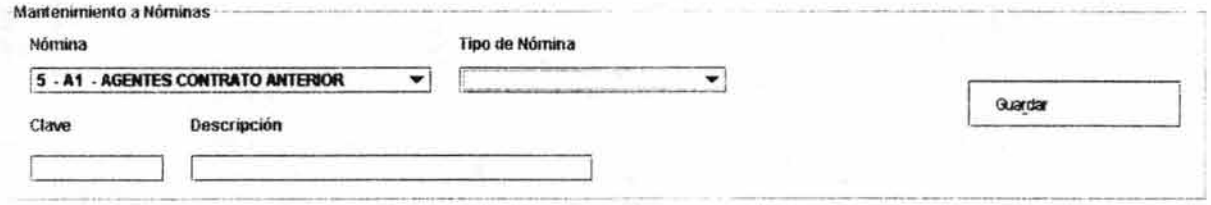


Figura 4.23
Mantenimiento a Nómina.

4.5 Implantación del Sistema.

Actualmente el sistema no se ha implantado en la empresa Sección Amarilla, ya que niveles directivos decidieron implantar el sistema SAP, el cuál integrará al departamento de Recursos Humanos y Contabilidad entre otros. Sin embargo el actual sistema puede personalizarse para las necesidades de cualquier empresa en el área de Recursos Humanos, incluyendo Nómina por supuesto.

Capítulo V
ESTRATEGIA Y TECNOLOGÍA

5.1. Herramientas CASE (Computer - Aided Systems Engineering)

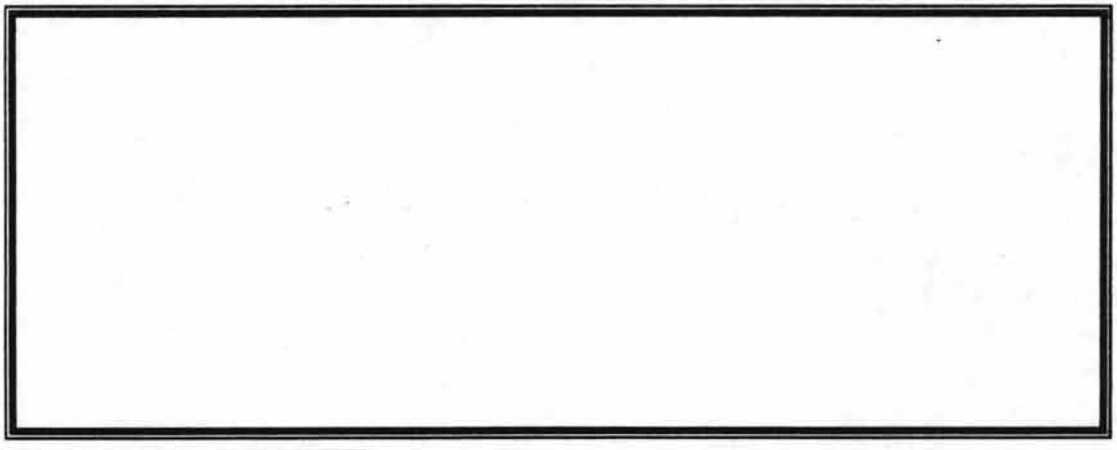
Las herramientas llamadas CASE, es la aplicación de tecnología informática las actividades, las técnicas y las metodologías propias de sistemas, estas apoyan a una o más fases del ciclo de desarrollo de un sistema. Es aquel software que utilizamos para diseñar la resolución de problemas planteados, en nuestro caso estas, se utilizaron para el diseño y parte del desarrollo de nuestro sistema. Estas herramientas nos facilitan el trabajo de diseño y desarrollo en el sentido que podemos llevar un mejor control de las diferentes etapas del desarrollo del sistema. Nos proporcionan la facilidad de guardar el análisis, diseño y desarrollo del sistema de una manera gráfica y consistente, así podemos lograr una visión más clara de nuestro trabajo.

La tecnología CASE proporciona la automatización del desarrollo de software, esto es de gran importancia en la ingeniería de software, esto se traduce en la disminución de tiempos para las diferentes etapas de desarrollo de un sistema de información.

Los elementos de las herramientas CASE:

- Diccionario de Objetos. Es dónde se almacenan los elementos construidos en la herramienta.
- Carga o descarga de los datos y de la información, son programas que nos facilitan la creación de estructuras de bases de datos o la transferencia de información de un SGBD a otro.
- Comprobación de errores, análisis de la estructura generada por la herramienta, lanzando como resultado inconsistencias, errores de lógica, etc.
- Interfaz de usuario, aquí se encuentran aquellas herramientas que nos facilitan un diseño de la interfaz de usuario mediante ventanas, íconos, en fin, un ambiente amable para el usuario.
- Generación de código fuente. Esta parte es muy importante ya que aporta un gran esfuerzo en el desarrollo del sistema de información, generando código fuente en el lenguaje que nosotros elijamos y claro que lo soporte la herramienta CASE.

En la página de Eduangi telecom43, nos presenta la estructura de las herramientas CASE:



[43] http://web.madritel.es/personales3/edcollado/index.html

En base a esta información que nos proporciona la página de Eduangi telecom podemos incluir como ejemplo en él CASE de alto nivel y CASE de bajo nivel a la herramienta CASE Visio, que nos permite realizar entre otras muchas cosas diagramas de flujo de análisis del sistema y diagramas de bases de datos. En el CASE cruzado de ciclo de vida podemos mencionar a Microsoft Project o bien Share point, los dos nos permiten la gestión de los proyectos y las actividades que existan en ellos.

5.1.1 Microsoft Visio

Esta herramienta nos proporciona diferentes instancias para acelerar nuestro proceso de desarrollo de nuestro sistema de información, hablando del desarrollo de este proyecto de Tesis, se utilizó la instancia para realizar el modelo conceptual de la base de datos y los diagramas de flujos de los procesos requeridos. Veamos la pantalla de Visio cuando se generó el modelo conceptual de nuestro sistema en la Fig 5.1

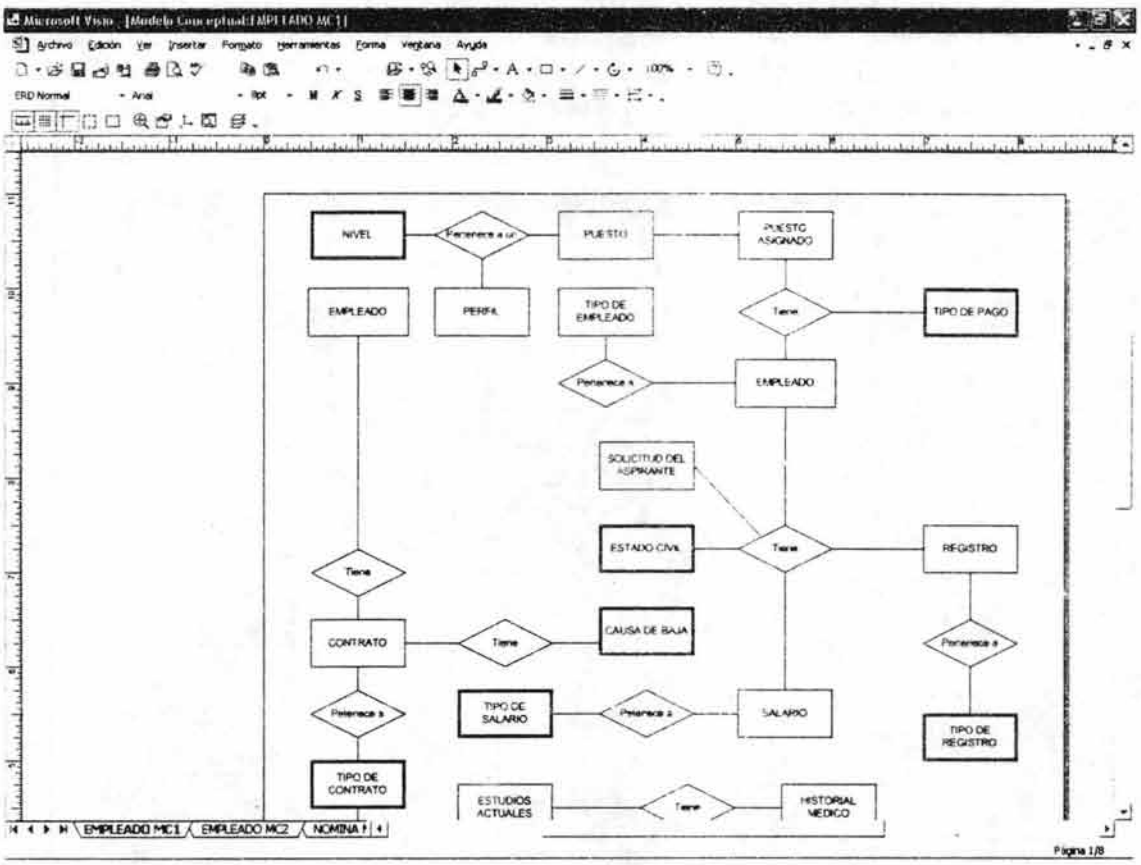


Figura 5.1
Pantalla Visio. Generando modelo Conceptual de datos.

Esta herramienta también nos permite realizar:

- Diagrama de Modelo de Base de Datos
- Diagrama de Bloques
- Diagramas de flujo
- Diagramas web
- Formularios
- Ingeniería de procesos

- Ingenieria eléctrica
- Ingeniería mecánica
- Organigramas
- Planos de Edificios
- Programación de proyectos
- Diseño de Redes
- Software.

5.1.2 Software Rational Rose

Esta herramienta CASE nos permite diseñar a detalle en un ambiente de objetos, las relaciones entre clases, incluyendo la plataforma del lenguaje en el cual serán programadas, este sofware en el ambiente de análisis del negocio utiliza UML (Visto en el capitulo I) generando Casos de uso. También nos permite realizar una ingenieria inversa a partir de código fuente JAVA, C++, Visual Basic, así como también la generación de estructuras especificando el SGBD de ORACLE, especificando todos los objetos y el esquema en que se va a trabajar. Veamos la pantalla de racional Rose cuando se realizaron los diagramas de clases en la Fig 5.2.

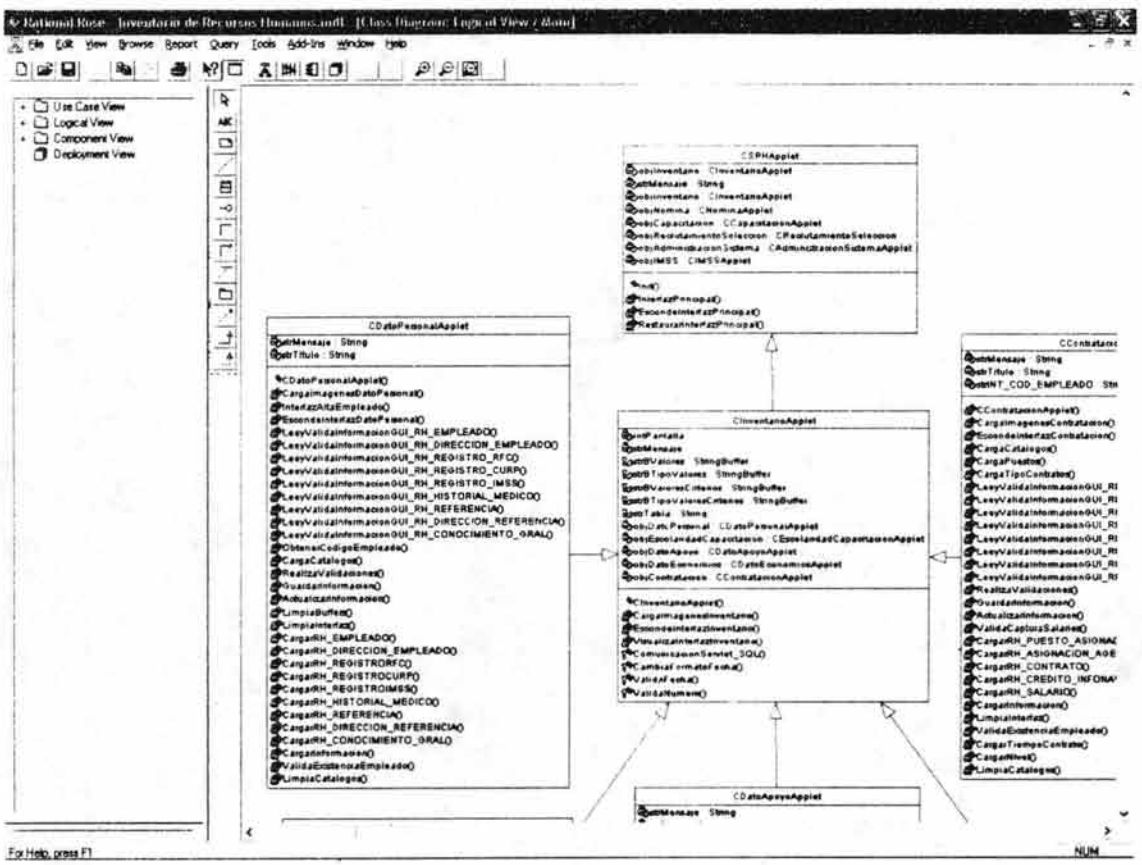


Figura 5.2
Pantalla Racional Rose. Generando jerarquía de clases base del sistema.

5.1.3 Software Power Designer

Esta herramienta CASE nos sirve para generar modelos conceptuales de datos y modelos físicos de base de datos, una gran ventaja de esta herramienta es que una vez generado el diagrama del modelo físico de la base de datos nos proporciona procesos dónde nos genera el Script correspondiente al diagrama de base de datos en la plataforma de SGBD que elijamos. También existe la ingeniería inversa en esta herramienta, de tal manera que a partir de un script podemos generar el diagrama de base de datos. En el diagrama podemos documentar la base de datos siendo así independiente del SGBD, la ingeniería inversa también funciona conectándose mediante un ODBC a una base de datos ya montada en algún servidor y generar el diagrama correspondiente. Veamos en la Fig 5.3 la pantalla de Power Designer cuando generamos el diagrama de base de datos.

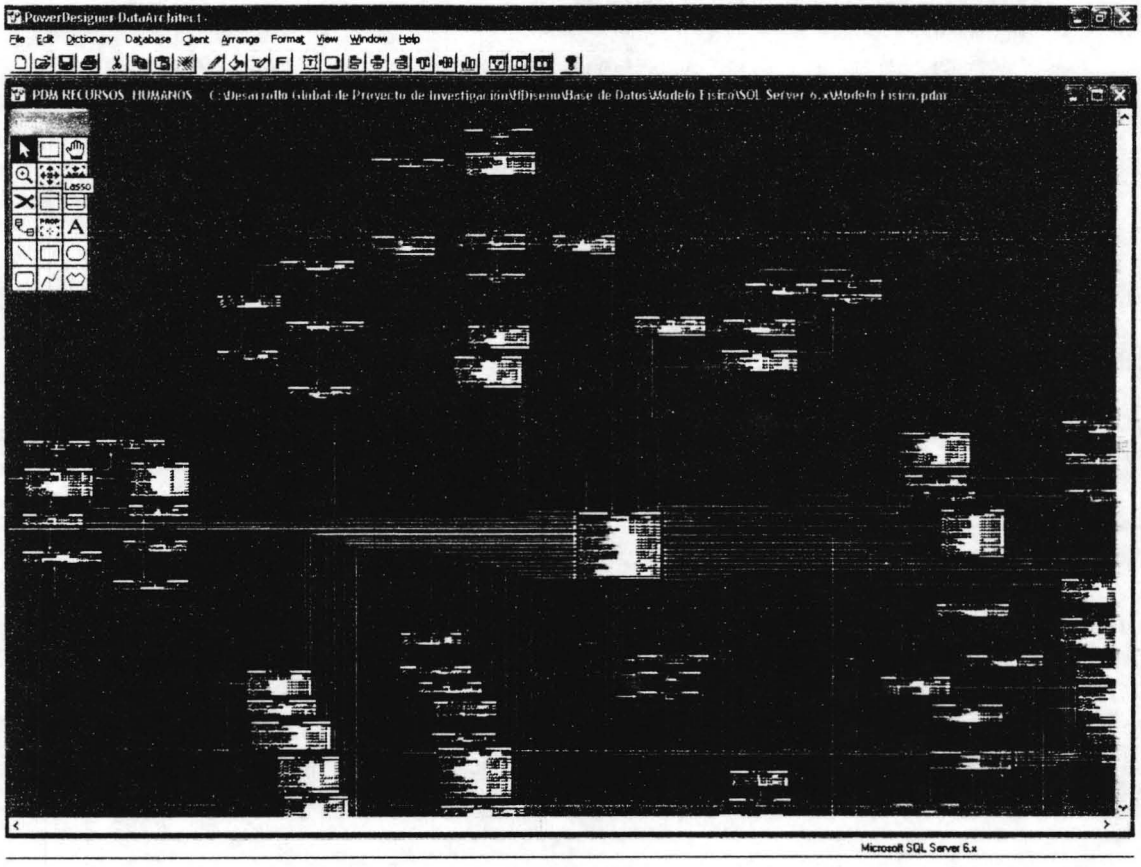


Figura 5.3
Pantalla Power Designer. Generando Modelo Físico de la Base de Datos del sistema.

5.2 Sistema Gestor de Base de datos SQL Server

SQL Server es un Sistema gestor de Base de Datos de Microsoft, que nos proporciona lo necesario para administrar Bases de datos, dentro de este manejador encontramos diferentes herramientas que nos permiten manipular la información de una manera organizada y eficiente. Entre otros programas encontramos el Administrador corporativo o bien Enterprise Manager, que contiene herramientas administrativas para diferentes componentes. Este cuenta con características como:

- Un motor de programación
- Funciones de alerta de administrador
- Operaciones de control entre múltiples servidores
- Administrar inicios de sesión
- Permisos a usuarios
- Crear secuencias de comandos
- Crear respaldos de bases de datos
- Administrar tablas, vistas, procedimientos almacenados, desencadenadores, índices, reglas, etc.

En fin hace lo que todo SGBD debe de cumplir. Este manejador como ya se ha mencionado en el desarrollo del presente proyecto, se utilizó para montar la base de datos diseñada, como también se utilizó para crear procesos utilizando recursos del servidor, mediante Procedimientos Almacenados. Un procedimiento almacenado es un programa que reside en la base de datos, y que cuando es invocado utiliza los recursos del servidor. Este programa contiene sentencias SQL, dónde la sintaxis cambia de manejador en manejador, por ejemplo, las sentencias utilizadas para crear un cursor en SQL Server, no son las mismas que se utilizan para hacerlo en ORACLE. Veamos como se crea un procedimiento almacenado en SQL Server:

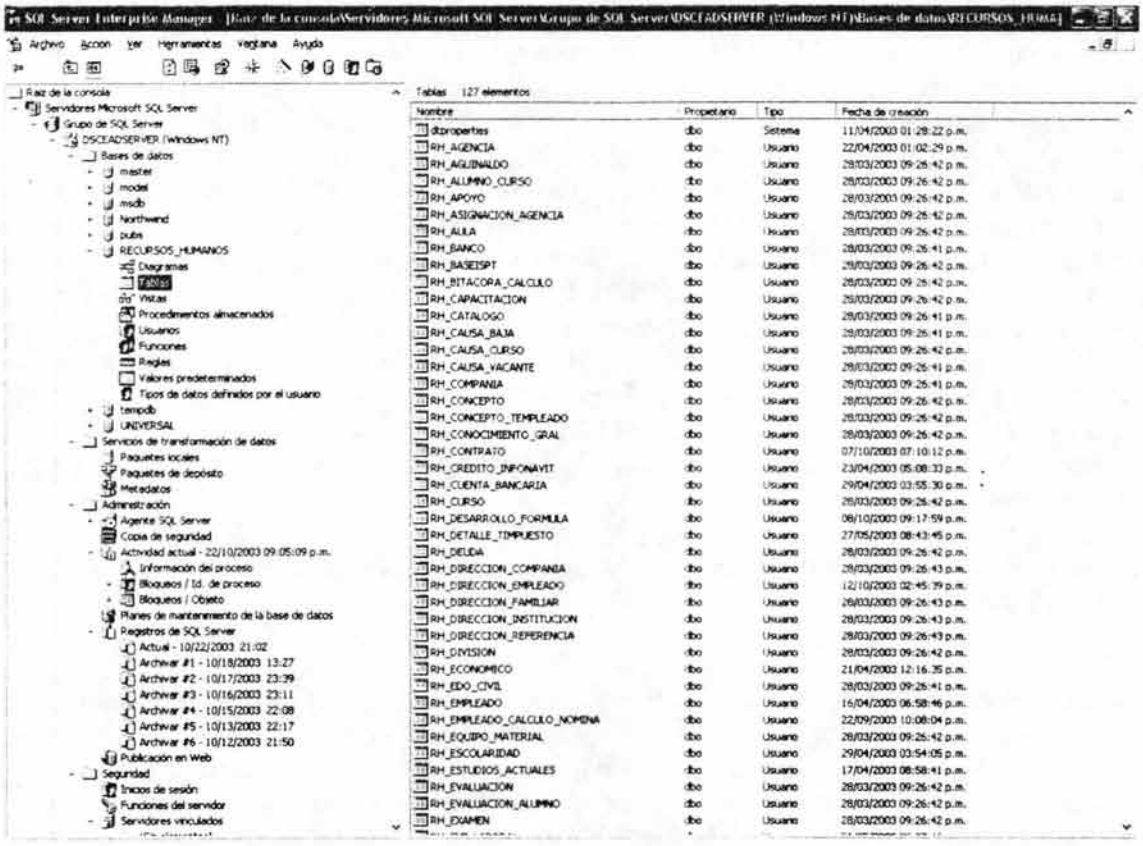
```
CREATE PROCEDURE RH_SP_CALCULO_DE_NOMINA
@SMI_COD_IDENTIFICADOR_NOMINA SMALLINT,
@SMI_COD_PERIODO SMALLINT
AS
DECLARE.... /* Declaración de variables*/
/*
Bloque de sentencias que se ejecutarán cada vez que se invoque al procedimiento
*/
```

Ahora veamos las sentencias para crear un Procedimiento almacenado en ORACLE.

```
CREATE PROCEDURE RH_SP_CALCULO_DE_NOMINA(
SMI_COD_IDENTIFICADOR_NOMINA IN NUMBER,
SMI_COD_PERIODO IN NUMBER,
DEC_IMPORTE_TOTAL OUT NUMBER)
IS
--Declaración de Variables;

BEGIN
-- Bloque de sentencias que se ejecutarán cada vez que se invoque al
procedimiento;
END RH_SP_CALCULO_DE_NOMINA;
```

Veamos ahora la pantalla de la Consola de Administración de SQL Server 7.0 en la Fig 5.4.



5.3 Lenguaje de Programación Orientado a Objetos JAVA

Java es un lenguaje de programación multiplataforma, esta característica fue la que lo hizo famoso, y cada vez mas gente está optando por este lenguaje. Es multiplataforma ya que es independiente del Sistema Operativo, es decir, los programas desarrollados en Java pueden ejecutarse en cualquier sistema operativo que tenga instalada la Máquina Virtual de Java. Por lo tanto su uso para sistemas en línea es muy práctico, ya que pueden acceder todas las máquinas con diferentes plataformas cada una. El Lenguaje Java fue desarrollado por Sun Microsystems, en un principio no tenía el objetivo de crear sistemas en Internet, sin embargo mas tarde se dieron cuenta que esta era la ruta que se le debía tomar en cuenta, siendo que en este lenguaje se comenzaron a programar páginas web que eran impensables en el común HTML Java tiene un amplio marco de programación, con esto quiero decir que se pueden dar solución a innumerables problemas planteados, y con la gran ventaja de que estos desarrollos pueden estar en la red Internet con un sistema de seguridad también programado en java, menciono esto por el hecho de que no tenemos que recurrir a otros lenguajes de programación para desarrollar módulos especificos con requerimientos especiales. Otra ventaja de este lenguaje es que es gratuito y podemos encontrarlo en la página de Sun; http://java.sun.com/, dónde también encontraremos documentación del mismo.

Dentro de java se han desarrollado diferentes tecnologías como J2EE, J2SE, J2ME, y cada desarrollador escoge la que más convenga a sus necesidades.

El lenguaje de programación es desarrollado por Sun Microsystems en el año de 1991, desarrollado en primer instancia con el objetivo de realizar software para aparatos electrodomésticos, sin embargo la compañía Sun se encuentra con un problema, ya que cada aparato cuenta con un dispositivo diferente, por lo tanto esta situación abrió un nuevo camino de investigación, y fue cuando surgió un nuevo lenguaje de programación independiente del dispositivo que fue nombrado como Oak, mas tarde en 1994 el grupo de investigadores de Sun miró hacia el navegador gráfico Mosaic para la Word Wide Web, y este les dio la idea de vincular sus investigaciones hacia el desarrollo de un lenguaje independiente de la plataforma y la arquitectura de los microprocesadores, ya que la idea original fue esta, la diferencia es que ahora lo harían para Internet y así fue a finales del año 1995 java se convirtió en el lenguaje de programación multiplataforma y también se incorporó una Máquina Virtual de java en netscape 2.0. Posteriormente en 1997 apareció java 1.1 y a finales de 1998 apareció java 1.2 (Java2), que según sus creadores esta es la primer versión profesional.

La sintaxis de Java deriva de la de C / C++, simplificándola y ampliándola, es decir, con las nuevas instrucciones de java podemos hacer mas en menos líneas de código.

Para traducir un programa escrito en Java, es necesario un compilador de java, este genera código intermedio (bytecode) y lo guarda en archivos CLASS, este lenguaje máquina no corresponde al del ordenador en el cual estemos trabajando, si no que corresponde a una máquina virtual, esta es una máquina que no existe, si no que es simulada por un programa, el cual se encarga de interpretar el código contenido en los archivos CLASS. De esta manera podemos correr nuestros programas en cualquier ordenador que tenga instalado la Máquina Virtual de Java. Para entenderlo mejor, veamos la Fig. 5.1.

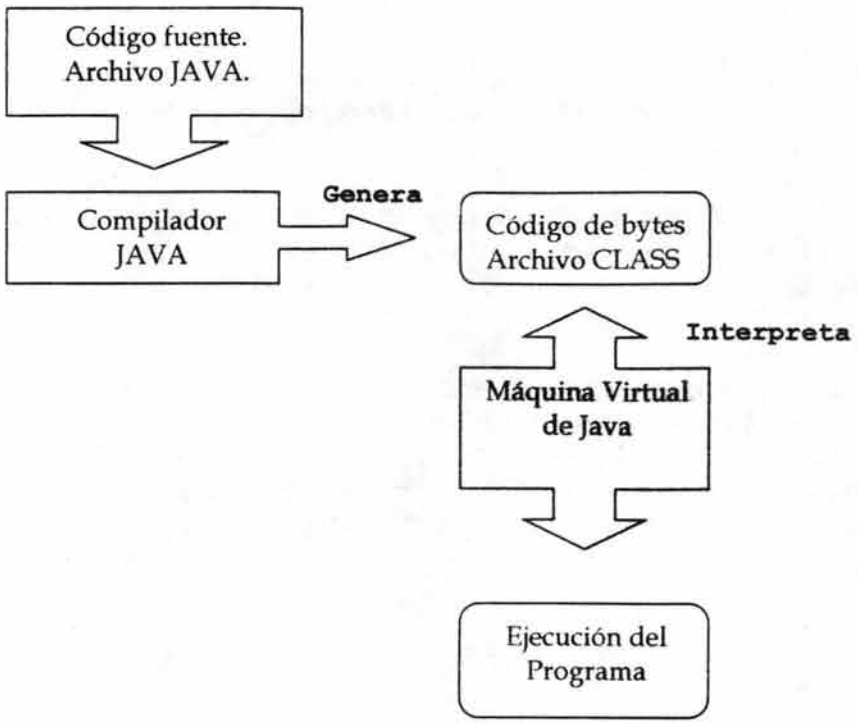


Figura 5.1
Flujo de ejecución de un programa en java.

5.3.1 Algunos Objetos de Java utilizados por el Sistema

StringBuffer

Este objeto lo utilizamos para guardar cadena de caracteres, en el caso del sistema, se utilizó para guardar valores de campos separados por un delimitador coma. Este objeto es modificable en tamaño y contenido, cuenta con métodos que lo permiten. Como:

append(Argumento g). Incrementa la cadena de caracteres, según la longitud del argumento.
delete(int a,int b). Elimina la cadena de caracteres, en la posición especificada, posición inicial;a, posición final; b.
length(). Retorna el número de caracteres que existen en la cadena.

Ejemplo.
StringBuffer strBCadena = new StringBuffer("CampoA");
StrBCadena.append(","); StrBCadena.append("CampoB");
Contenido de strBCadena: "CampoA,CampoB"
Valor retornado por strBCadena.length(): 13
Contenido de strBCadena después de strBCadena.delete(0,3); "poA,CampoB"

Vector

Este objeto se utilizó como un contenedor de objetos, en este caso de StringBuffers o String, se utilizaron métodos como:

add(int indice,Objeto). Se encarga de agregar a los objetos en el índice especificado.
removeAllElements(). Se encarga de vaciar al vector.
size(). Retorna el número de objetos existentes en el vector.

Ejemplo:
objVector.add(0,new Integer(intInstruccionSQL));
objVector.add(1,strTabla.toString());
objVector.add(2,strBValores.toString());

Valor de retorno por objVector.size(): 3

Interface Enumeration

Para obtener los elementos de un vector, recurrimos a la interface Enumeration. Una interface como lo menciona Javier Cevallos44 : Es un dispositivo que permite interactuar a objetos no relacionados entre sí. El Enumeration nos permite acceder a elementos de una estructura en forma secuencial, posee dos métodos:

hasMoreElements(). Retorna un bolean e indica si existen más elementos en la colección, este caso el Vector.

nextElement(). Retorna el siguiente objeto contenido en la colección.

Estos métodos están definidos por la clase VectorEnumerator que implementa el interface Enumeration. Por otra parte, la clase Vector posee el método elements() que regresa un objeto VectorEnumerator. Por lo tanto para obtener los elementos del vector mencionado anteriormente y guardarlos en StringBuffers, escribimos el siguiente Ejemplo.

```
Enumeration enum = objVector.elements();
StringBuffer strBElemento = new StringBuffer();
```

44 En su libro Java 2 Curso de Programación página 371 INTERFACES.

```
while(enum.hasMoreElements())
{
        strBElemento.append( (String)enum.nextElement());
}
```

StringTokenizer

Este objeto se encarga de obtener los campos (tokens) de un StringBuffer, uno por uno, especificando el delimitador. Esta clase también implementa la interface, sin embargo utilicé los métodos equivalentes miembros de la clase StringTokenizer hasMoreTokens() y nextToken(). Algunos de los métodos que posee esta clase son:

hasMoreTokens(). Retorna un bolean e indica si existen más tokens.
nextToken(). Retorna un String, y es el siguiente token que encuentra después de cada delimitador.
countTokens(). Retorna el número total de tokens.

Por lo tanto, para obtener los tokens del StringBuffer mencionado anteriormente, escribimos el siguiente código:

```
StringTokenizer StrTkCampos = new StringTokenizer(strBElemento.toString(),",");
String strCampo = new String();

while(StrTkCampos.hasMoreTokens())
        strCampo = StrTkCampos.nextToken();
```

5.3.2 Applets

Los applets no son mas que programas realizados en java que se ejecutan en un Browser de Internet que soporte Java, y estos se ejecutan siempre en el cliente. Estos applets se almacenan en servidores Web y se cargan en el cliente a través de la red, no necesitan de la red mientras se ejecuta, siempre y cuando el applet no realice una conexión con un servlet, o por medio de JDBC u otra instancia, no tienen acceso a los archivos y dispositivos del cliente. Un applet es un archivo CLASS compilado, este es interpretado por el navegador que cuenta con la Máquina Virtual de Java en cada cliente, veamos el esquema en la Fig. 5.2. En la Fig. 5.3 podremos apreciar el ejemplo de un applet sencillo, en este caso vemos que hereda de la clase JApplet, esto es por utilizamos componentes swing, para una interfaz gráfica, ya que comúnmente una clase se vuelve applet al heredar directamente de la clase Applet.

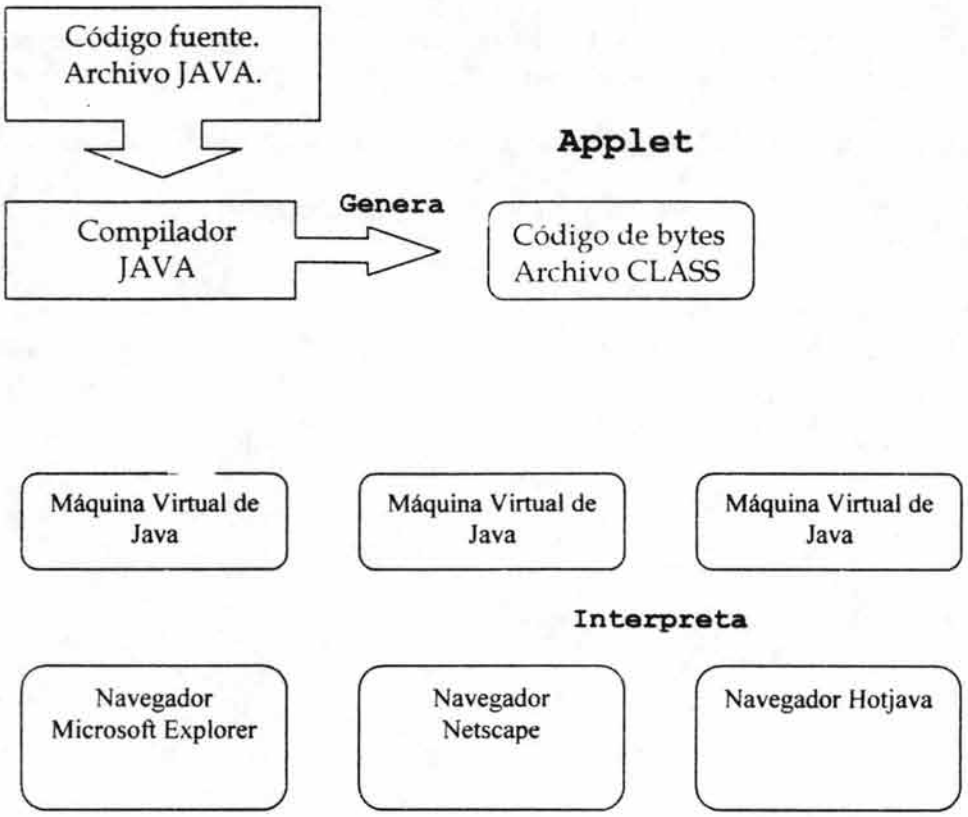


Figura 5.2
Flujo de ejecución de un programa en java Applet.

Un applet se compone aunque no necesariamente se tienen que definir de los métodos que a continuación se presentan, este es el ciclo de vida de un applet.

init(). Este método se ejecuta automáticamente solo una vez, al cargar el applet.
start(). Este método se ejecuta automáticamente después del método init().
paint(). Es llamado cada vez que es necesario repintar el contenedor del applet.
stop(). Este método informa al applet que su ejecución va ser detenida.
destroy(). Este informa al applet que va ser destruido, esto es para que libere cualquier recurso utilizado y se ejecuta después del método destroy().

```
import java.awt.*;
import java.applet.*;
import java.util.*;

public class CClaseApplet extends JApplet
{
        private strMensaje;
        public void init()
        {       strMensaje = "Método init()";
        }
        public void start()
        {       strMensaje = "Método start()";
        }
        public void paint(Graphic g)
        {       //Código para mostrar en pantalla
        }
        public void stop(Graphic g)
        {       strMensaje = "Método stop()";
        }
        public void destroy(Graphic g)
        {       strMensaje = "Método destroy()";
        }
```

Figura 5.3
Ejemplo de un Applet

5.3.3 Servlets

Básicamente un Servlet no es más que un programa escrito en Java que se ejecuta en el servidor, y puede generar dinámicamente páginas web. Por lo tanto, estos residen en el servidor y cuando son llamados por primera vez, estos se cargan en memoria indefinidamente por el **motor de servlets** el cual administra la carga y descarga de los servlets y como se ejecutan en el servidor, a diferencia de los applets que se ejecutan en una máquina virtual proporcionada por navegador de cada máquina, con una probable incompatibilidad, los servlets se ejecutan en la máquina virtual contenida en el servidor, con esto quiero decir que su ejecución es independiente del navegador de cada máquina.

Ciclo de vida de un servlet

Cuando el motor de servlets carga al servlet, se ejecuta el método init(), como el servlet se carga solamente una vez en memoria recibiendo peticiones, el método init() se ejecuta solamente una vez, y después manda a llamar a su constructor. Posteriormente se ejecuta el método service() para manejar las peticiones que reciba el servlet con subprocesos diferentes. La mayoría de los servlets, están diseñados para trabajar en un entorno HTTP, por lo tanto existen métodos especializados a las peticiones HTTP::

doGet. Maneja las peticiones del tipo GET.
doPost. Maneja las peticiones del tipo POST.

El motor de servlets construye los objetos:

HttpServletRequest. Contiene información de la petición, así como la información enviada por el cliente.
HttpServletResponse. Contiene la respuesta del servlet, así como la información que se le enviará al cliente que realizó dicha petición.

Por último la llamada al método destroy(), significa que el motor de servlets notifica a cada servlet que será descargado, siendo este el que se encarga de descargar el servlet, y no el método destroy().

Veamos el esquema de operación del servlet en la Fig. 5.4

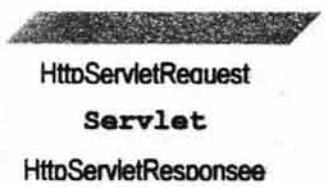


Figura 5.4
Esquema de operación

Un servlet puede sincronizar las peticiones que se hacen concurrentemente, reenviar peticiones a otros servlets y a otros servidores. De esta manera los servlets puedes distribuir la carga en varios servidores. Los servlets son capaces de servir concurrentemente a varios clientes.Los métodos en el servlet accesan a información compartida, dichos métodos pueden sincronizar las acciones de los servlets para tener

acceso exclusivo a los recursos o crear un servlet que atienda solo una petición de cliente a la vez, para esto el servlet debe de implementar la interfaz SingleThreadModel.

Ahora veamos en la Fig. 5.5 el ciclo de vida de un servlet

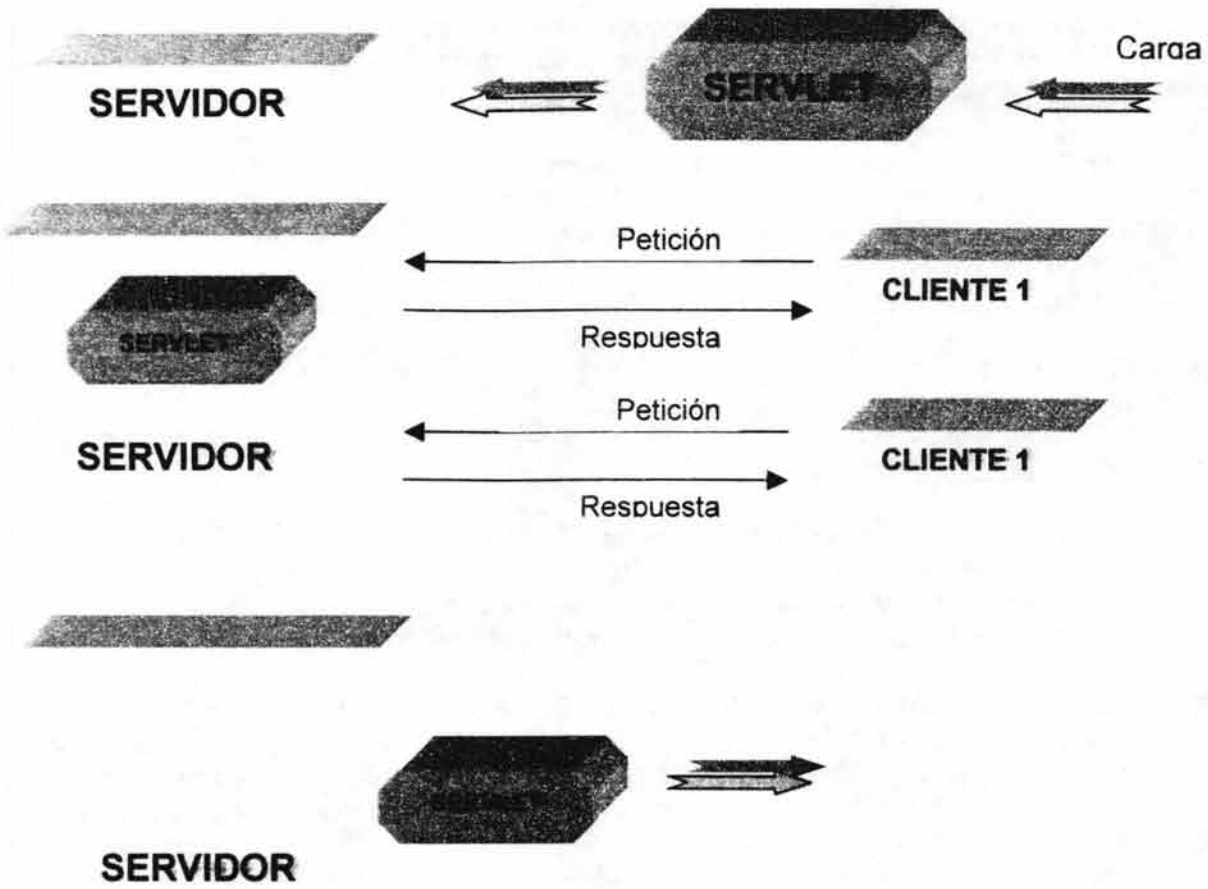


Figura 5.5
Ciclo de Vida del Servlet

5.3.4 Comunicación Applet – Servlet

Como ya hemos visto a lo largo del desarrollo de nuestro sistema, para realizar la Interfaz Gráfica de Usuario, se utilizó componentes swing en Applets, los cuales recibe la información directa del usuario, el applet también es el encargado de realizar una conexión con el servlet, basado en una comunicación HTTP tunneling. Utilizando esta comunicación un applet puede comunicarse con varios servlets, y estos a su vez pueden comunicarse con varios servidores y servlets, por lo tanto, a partir de un applet podemos enviar información a varios servidores con una sola petición.

El HTTP tunneling es el proceso de crear un subprotocolo encima del protocolo HTTP. El subprotocolo contiene información que es requerida por un objeto en el servidor, es decir, un objeto del servlet HttpServletRequest, este subprotocolo tiene como tarea específica transportar información entre el cliente y el servidor. Esta comunicación puede estar basada en texto o bien en objeto.

Comunicación http basada en texto

Para que sea posible esta comunicación, el applet debe de abrir una conexión al servlet, especificando la URL del servlet. Veámoslo en siguiente ejemplo:

```
String strBaseURL = "http://localhost:8080/servlet/CInventarioServlet";
String strParametros = "cod_Empleado=3128&cod_Puesto=3";
URL direccion = new URL(strBaseURL + " ? " + strParametros);
URLConnection servletConnection = direccion.openConnection();
```

Como podemos apreciar, en la petición http por el método GET, los parámetros viajan en la URL, la cual va como parámetro al constructor del objeto URL, a su vez, se crea un objeto URLConnection que es el que guardará la conexión abierta al servlet especificado, en este caso llamado CinventarioServlet.. Del lado del servidor, es decir en el servlet, se reciben los parámetros, estos están contenidos en el objeto HttpServletRequest req, y con el método getParameterNames() y getParameterValues(), se leen los nombre de los parámetros y sus valores correspondientemente mediante un Enumeration explicado en el punto 2.3.1 de este capítulo, posteriormente en este ejemplo, el método ObtenerInformacion se encarga de conectarse a la base de datos y retornar un StringBuffer como resultado de una consulta, que es el nombre completo del empleado, esta es la respuesta que tiene que dar el servlet al applet utilizado veamos el código fuente:

```
Enumeration nombresParams = req.getParameterNames();
   while (nombresParams.hasMoreElements())
   {
    String strCodigoEmpleado = (String)nombresParams.nextElement();
    String strCodigoPuesto = req.getParameterValues(nombre)[0];
    }
        //private StringBuffer ObtenerInformacion(String,String)
        StringBuffer strBInformación = ObtenerInformacion(strCodigoEmpleado, strCodigoPuesto)
```

Una vez que poseemos la información deseada, hay que responder al applet utilizando el objeto HttpServletResponse, y ya que estamos en una comunicación basada en texto, a este objeto se le notifica el tipo de rspuesta con el método setContentType(), posteriormente se crea un objeto PrintWriter para enviar la respuesta al servlet, en este caso la respuesta será una pequeña página html, dónde el applet será el encargado de mostrarla con el método miApplet.getAppletContext.().showDocument(URL Direccion).

```
// Tipo de la respuesta que será enviada al cliente
```

```
res.setContentType("text/html");
// Obtener un 'PrintWriter' para devolver una respuesta
   // al solicitante
   PrintWriter solicitante = res.getWriter();

         // Responder al solicitante
         solicitante.println("<html>");
         solicitante.println("<title>Respuesta a la solicitud</title>");
         solicitante.println("Nombre:<BR>" + strBInformación.toString());
         solicitante.println("</html>");
```

Ahora bien, el enviar la respuesta en un html implica que el applet suspenda su ejecución, ya que el explorador a cambiado de página, si no se desea esto podemos utilizar el siguiente código para que el mismo applet sea el encargado de imprimir la información obtenida en pantalla, simplemente en lugar de escribir el código HTML en el println, solamente escribimos la información que se le enviará al applet, lo que en realidad cambia es la forma de recibir la información por parte del applet.

```
         // Responder al solicitante
         Solicitante.println(strBInformación.toString());
```

Y del lado del applet, el método getInputStream() de nuestro objeto servletConnection nos retorna un objeto InputStream, posteriormente se crea un BufferedReader para obtener la información enviándole como parámetro en su constructor el InputStream obtenido, posteriormente se manda a llamar al método readLine() del objeto BufferedReader que nos retorna la información que hay en él, guardándola en un arreglo de String's.

```
InputStream input = servletConnection.getInputStream();
BufferedReader br = new BufferedReader(new InputStreamReader(input));

Int intContador = 0;
String strAInformacion[] = new String[4];

         While((strAInformacion[i] = br.readLine()) != null)
                  IntContador ++;
br.close()
```

Comunicación http basada en Objeto

Ahora veremos la transferencia de información, ahora encapsulada en un objeto. En el Applet se crea un objeto que nos servirá como contenedor de objetos, es decir este contendrá toda la información que deseamos enviarle al Servlet, a diferencia del método GET que la información viaja en el URL, aquí irá encapsulada en un objeto, el procedimiento al inicio es el mismo que vimos en la comunicación basada en texto, pues bien, una vez que tenemos el objeto conexión, le informamos que se enviarán información y que también se recibirá, también indicamos el tipo de contenido que se manejará con el método setRequestProperty(), el tipo application/octet-stream nos permite enviar datos binarios.

```
objVector.add(1,strInformacion.toString());
//La conexión es informada que aceptará datos de retorno del servlet
servletConnection.setDoInput(true);
//La conexión es informada que enviará información al Servlet
servletConnection.setDoOutput(true);
//Añadir las cabeceras de HTTP
```

servletConnection.setRequestProperty("Content-Type","application/octet-stream");

Ahora preparado el terreno para enviar la información, nos disponemos a enviarla, ¿qué se enviará?, los objetos, y a esto se llama seriación, esto lo logramos utilizando el método writeObject() de la clase ObjectOutputStream.

```
ObjectOutputStream objOuputSSalida = new ObjectOutputStream(
                                                servletConnection.getOutputStream());
        //Mandar los objetos al servlet por el flujo de salida
        objOuputSSalida.writeObject(objVector);
        //Asegurar que envíe el contenido forzando el vaciado del buffer
        objOuputSSalida.flush();
        objOuputSSalida.close();
```

Ahora nos trasladamos al Servlet, este debe ser el encargado realizar la deseriación del objeto enviado por el Applet, así se le llama a la operación de leer los objetos recibidos, esto lo logramos con el método readObject() del objeto ObjectInputStream, no sin antes crear este a partir del método getInputStream() del objeto HttpServletRequest, realizamos una conversión de datos, para obtener la información a partir de un Vector, teniendo este objeto ya es sencillo obtenerla con un Enumeration, como ya se explicó anteriormente en este capítulo.

```
ObjectInputStream input=new ObjectInputStream(req.getInputStream());
        Vector objVector = (Vector)input.readObject();
```

Continuando en el servlet, este también regresará la información encapsulada, por lo tanto, tenemos que informar al objeto HttpServletResponse el tipo de respuesta que se enviará, en este caso será application/x-java-serialized-object, el cual indica la seriación de un objeto, se crea un ObjectOutputStream que será nuestro flujo de salida, y al igual que el applet utilizó el método writeObject(), el servlet también lo hace, para realizar una seriación de objeto.

```
response.setContentType("application/x-java-serialized-object");
        //Abrir el flujo de salida para enviar objetos
        ObjectOutputStream objOuputSSalida = new ObjectOutputStream(response.getOutputStream());
        objOuputSSalida.writeObject(objVector);
        //Asegurar que envíe el contenido forzando el vaciado del buffer
        objOuputSSalida.flush();
        objOuputSSalida.close();
```

De la misma forma que hizo el servlet para realizar una deseriación del objeto, lo hace el applet:

```
ObjectInputStream ObjInputSEntrada = new ObjectInputStream(
                                        servletConnection.getInputStream());
        objVector = (Vector)ObjInputSEntrada.readObject();
```

y de esta manera, utilizando la seriación de objetos, nos es posible enviar información Applet – Servlet – Applet. Veamos de una manera gráfica estas operaciones en la Fig. 5.5

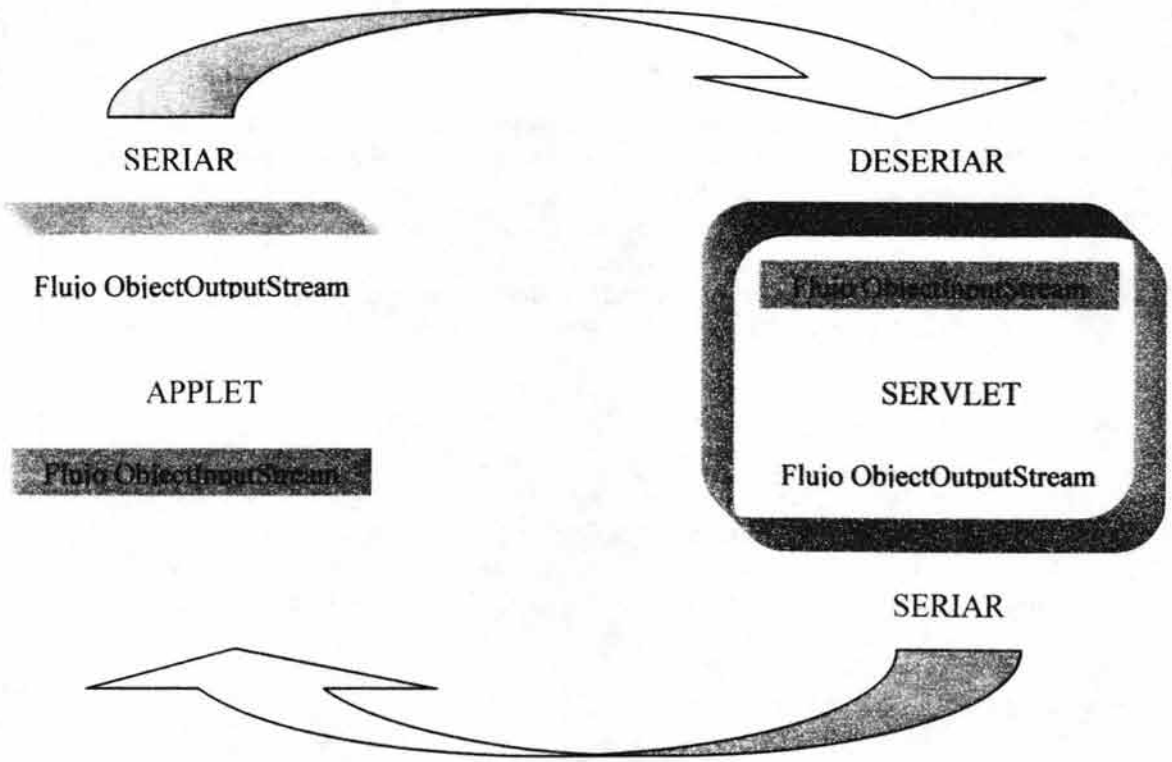


Figura 5.6
Seriación de Objetos en
Una comunicación Applet – Servlet - Applet

5.3.5 Acceso a Base de Datos con JDBC

JDBC es una tecnología de Java que nos permite tener acceso a gran variedad de sistemas gestores de bases de datos, es decir esta es una interfaz entre los programas realizados en Java y los SGBD45 , por decirlo de otra manera es una API46, mediante una clase de Java conocida como controlador JDBC. El controlador llamado puente JDBC – ODBC hace posible que JDBC pueda acceder a todos los controladores de ODBC 47.

Bien, para realizar una conexión a la base de datos, primero debemos de cargar la clase del controlador JDBC mediante el método forName(), que recibe como parámetro el nombre del controlador, en este caso es el JDBC – ODBC:

```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

Posteriormente se abre una conexión a la base de datos, mediante el método getConnection() de DriveManager que es la interfaz que registra los controladores JDBC, que recibe como parámetros un url que es una especificación del driver y el nombre del ODBC, en este caso llamado RECURSOS_HUMANOS, los dos siguientes parámetros son el usuario y la contraseña a nuestra base de datos.

```
String url = "jdbc:odbc:RECURSOS_HUMANOS";
Connection con = DriverManager.getConnection(url, "sa", "");
```

Una vez que tenemos una conexión abierta a una base de datos, podemos realizar todas las operaciones posibles con el lenguaje SQL de base de datos. Para obtener información, utilizamos un objeto Statement que se encarga de ejecutar cadenas SQL, y este lo obtenemos con el método createStatement() del objeto Connection, posteriormente ejecutamos una sentencia SQL con el método executeQuery() del objeto Statement, el resultado lo almacenamos en un Resultset, y obtenemos de este la información los los métodos correspondientes:

```
stmt = con.createStatement();
strSentencia = " SELECT strNombre, strAppaterno FROM rh_empleado ";
ResultSet  rsInformacion = stmt.executeQuery(strSentencia);
while(objInventario.rsInformacion.next())
{
        strBRegistro = new StringBuffer(objInventario.rsInformacion.getString(1) + ","
                        + objInventario.rsInformacion.getString(2));
        objVector.add(intRegistro,strBRegistro.toString());
        strBRegistro.delete(0,strBRegistro.length());
        intRegistro ++;
}
```

Para realizar actualizaciones del tipo INSERT, UPDATE Y DELETE, utilizamos el método executeUpdate(), así como instrucciones de definición de datos CREATE TABLE, CREATE INDEX, devuelve un contador con el número de registros actualizados, el procedimiento ejecutar y recibir los resultados es el mismo, la diferencia es que los resultados los podemos guardar en una variable numérica como double. Veámoslo de una manera gráfica en la Fig. 5.7.

---

45 Sistemas Gestores de Bases de Datos

46 Application Interface. Interfaz de Aplicación

47 Open Data Base Connectivity. Conectividad a Base de datos Abierta

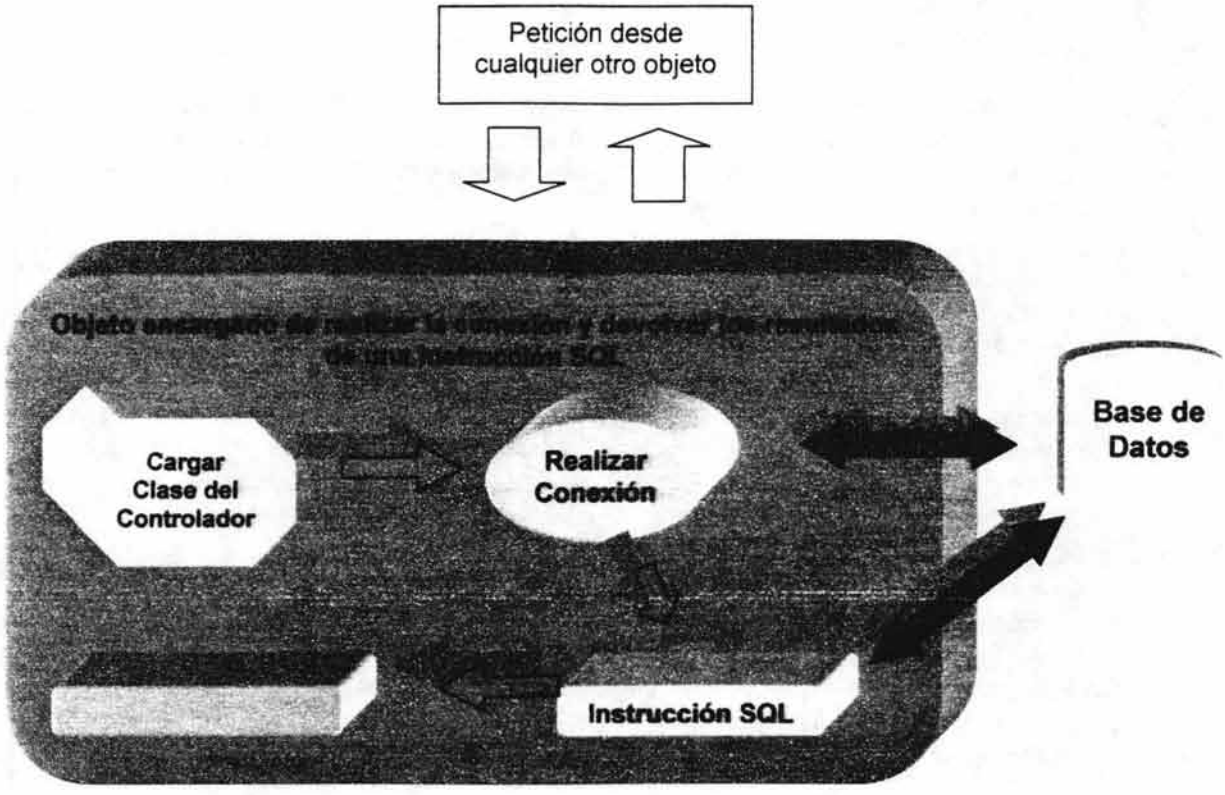


Figura 5.7
Proceso de Acceso a la BD.

5.4 Interfaz Gráfica de Usuario Componentes Swing

Los componentes Swing pertenecen a la JFC(Java Foundation Classes), es el paquete gráfico que apareció en la versión 1.2 de java, este está compuesto por un gran conjunto de componentes de interfaces de usuario que funcionan en el mayor número de plataformas. Cada uno de estos componentes pueden presentar diversos aspectos y comportamientos.

Originalmente java cuenta con un paquete de interfaces de usuario llamado AWT (Abstract Window Toolkit), Swing viene a revolucionar esta interfaces, proporcionándonos componentes más potentes y dinámicos. Algunas de las ventajas que representa Swing respecto a su antecedente AWT son:

- Arquitectura Modelo-Vista-Controlador: Esta arquitectura da lugar a todo un enfoque de desarrollo muy arraigado en los entornos gráficos de usuario realizados con técnicas orientadas a objetos. Cada componente tiene asociado una clase de modelo de datos y una interfaz que utiliza. Se puede crear un modelo de datos personalizado para cada componente, con sólo heredar de la clase *Model*.
- Contenedores anidados: Cualquier componente puede estar anidado en otro. Por ejemplo, un gráfico se puede anidar en una lista.
- Escritorios virtuales: Se pueden crear escritorios virtuales o "interfaz de múltiples documentos" mediante las clases *JDesktopPane* y *JinternalFrame*, de hecho así se realizó la interfase de nuestro sistema.
- Diálogos personalizados: Se pueden crear multitud de formas de mensajes y opciones de diálogo con el usuario, mediante la clase *JOptionPane*.
- Componentes para tablas y árboles de datos: Mediante las clases *JTable* y *JTree*.

Veamos en la siguiente figura una de las ventanas de nuestro sistema, claro, realizada con componentes Swing:

Figura 5.8
Ventana realizada con Componentes Swing
Mantenimiento de Fórmulas

5.5. Infraestructura

5.5.1 La red de Computadoras

No es objetivo de este proyecto explicar todo acerca de redes, solo se expondrá lo básico que se utilizó para implantar nuestro sistema.

Una red es un conjunto de computadoras autónomas interconectadas entre si, esto con el fin de compartir recursos y facilidad de comunicación. En nuestro caso la red nos servirá para que todos los clientes que así lo requieran, puedan acceder al sistema que reside en un Servidor el cual se encargará de almacenar al sistema como tal y a la información utilizada por este.

En los últimos años las redes de computación han tenido una gran importancia, ya que las empresas se han valido de ellas para montar sus aplicaciones como correo electrónico, y las funciones para las operaciones del negocio, como nuestro sistema.

Existen diferentes tipos de redes, se utiliza cada una de ellas dependiendo de las necesidades del negocio y los requerimientos del o los sistemas que se vayan a implantar.

Como bien lo menciona el Manual de Redes & Transmisión de Datos Creado por Softdownload Argentina, las redes se pueden clasificar de acuerdo a las dimensiones de la tecnología de transmisión y del tamaño.

Tecnologías de transmisión:
Broadcast. Un solo canal de comunicación compartido por todas las máquinas. Un paquete enviado por alguna máquina es recibido por todas las otras.
Point to point. Varias conexiones entre pares individuales de máquinas. Los paquetes de A a B pueden atravesar máquinas intermedias, entonces se necesita el ruteo (routing [e3]48) para dirigirlos.

5.5.2 Red de Área Local

Redes de Área Local (Local Area Networks o LANs) son aquellas que usualmente están confinadas a un área geográfica específica como un edificio o una Universidad, sin embargo no son necesariamente pequeñas, estas pueden estar compuestas de un gran diseño que albergará cientos de computadores con miles de usuarios.

5.5.3 Red de área amplia

La interconexión de áreas amplias ("Wide Area Networks) es la conexión de múltiples LAN´s que se encuentran geográficamente separadas. Esto se logra utilizando servicios que incluyen líneas de teléfono dedicadas (punto a punto) o vínculos satelitales. Estas pueden ser tan simple como proveer a los clientes de modems para permitir el acceso a un servidor remoto49 .

[48] Routing. Acción de elegir una ruta por la cual se enviarán los datos recibidos.
[49] Remoto. Elemento que se encuentra geográficamente en otro sitio diferente al del trabajo.

5.5.4 Internet

La Internet estuvo inicialmente restringida a instituciones militares y académicas. Nació siendo la red de la Administración de Programas de Investigación Avanzados (Advanced Research Programs Administration) del Departamento de Defensa de los Estados Unidos de Norteamérica y se le conoció como ARPANet.

La internet es una red de redes a nivel mundial, dónde los usuarios pueden compartir información, la comunicación en internet es posible entre redes de diferentes ambientes, esto se ha logrado gracias al desarrollo de protocolos de comunicación. Los protocolos de internet que trabajan en conjunto para la transmisión de datos son
Transmisión Control Protocol (TCP)
Internet Protocol

5.5.5 Intranet

Una LAN o una WAN que utiliza las tecnologías de internet es llamada intranet, esta brinda a los usuarios la capacidad de compartir recursos internos dinámicamente. Para utilizar una intranet, las computadoras necesitan:
TCP/IP instalado
Un navegador Web instalado
Un servidor Web instalado (únicamente en la máquina que será el servidor)

5.5.6 Protocolos de Comunicación

Los protocolos son un conjunto de reglas para el intercambio de información que nos permite comunicarnos entre diferentes redes.

Los Protocolos de red son estándares que permiten a las computadoras comunicarse.
Un protocolo define:
Como las computadoras se identificarán unas a otras sobre una red.
La forma que los datos deben tomar para ser transmitidos.
Como esta información debiera ser procesada una vez que llega a destino.

5.5.7 Hardware

El [EHC4]hardware dónde se implantará el sistema es:

Servidor HP con 2 procesadores intel
Disco duro de 500 Gb
Memoria RAM de 10 Gb

5.5.8 Software

El [EHC5]software que se utilizará para implantar el sistema es:

Sistema Operativo:
Servidor de Base de Datos: Microsoft SQL Server 2000
Servidor Web: Servidor Apache 1.3.22

GLOSARIO

- ✓ Sistema. Conjunto de elementos que actúan entre sí para lograr un objetivo en común,
- ✓ Cronograma. Documento dónde se planean las actividades de un proyecto, esto es por fechas, duración de cada actividad, etc.
- ✓ Muestreo. Tomar solo una muestra de todo un universo de elementos.
- ✓ Código fuente. Conjunto de instrucciones en un cierto lenguaje de programación.
- ✓ Clase. Conjunto de n objetos con características propias y heredadas.
- ✓ Objeto. Entidad que posee conocimiento de tareas y de cómo realizarlas.
- ✓ Herencia. Acción de heredar características de una clase superior.
- ✓ Semántica de datos. Implementación de las reglas de negocio en un diseño de BD.
- ✓ Integridad de la información. Que toda la información contenida en las entidades concuerde con las relaciones establecidas de acuerdo a ciertas reglas.
- ✓ **Ocurrencias.** Son los valores que tomará un campo.
- ✓ **Sistema.** Conjunto de elementos relacionados entre sí para cumplir un objetivo general.

BIBLIOGRAFÍA

ADORACIÓN DE MIGUEL Y MARIO PIATTINI (1999). " Fundamentos y Modelos de Bases de Datos " Universidad Carlos III de Matrid, 2ª Edición, Alfaomega Grupo Editor S.A de C:V.

KENNETH E. KENDALL Y JULIE E. KENDALL (1997) " Análisis y Diseño de Sistemas " Rutgers University, School of Business-Camdem, New Jersey USA, 3ª Edición, Pearson Educación

ING ERNESTO PEÑALOZA ROMERO (1994) " Fundamentos de Programación " Universidad Nacional Autónoma de México 2ª Edición, Unidad de Extensión Universitaria, Coordinación de Ediciones.

FRANCISCO JAVIER CEVALLOS (2000) " Java 2 Curso de Programación " Escuela Politécnica Universidad de Alcalá Madrid, España, Edición Original, Alfaomega Grupo Editor S.A de C:V.

DAVID M. GEARY (1999). " Graphic Java Mastering the JFC Volume II Swing " The Sun Microsystems Press Java Series 3RD Edition Sun Microsystems.

Suresh Rajagopalan, Ramesh Rajamani, Ramesh Krishnaswamy, and Sridhar Vijendran (2002). " Java Servlet Programming Bible " Hungry Minds, Inc.

Global Knowledge Curso "Java Enterprise Server Programing Servlet " (D.F 2003) terminado el 31 de Enero de 2003

PHIL HANNA (2002) " JSP Manual de Referencia " Traducido de la Primera Edición JSP: The Complete Referente, McGRAW-HILL/INTERAMERICANA DE ESPAÑA, S.A. U.

Eduangi telecom: **http://web.madritel.es/personales3/edcollado/index.html**. Es una web en donde se expone a la comunidad de internet una serie de referencias tecnológicas. Fundada en 1997, Madrid España.

Alberto Fernandez (2001) " Manual de Intranet " Universidad de Guadalajara, consulta desde http://www.softdownload.com.ar