UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SISTEMA DE CONTROL BIBLIOTECARIO DE USUARIOS

T E S I S

QUE PARA OBTENER EL TÍTULO DE

INGENIERO EN COMPUTACIÓN

P R E S E N T A N

HERNÁNDEZ REYES DELFINO

LORENZANA GUTIÉRREZ PABLO ANTONIO

PÉREZ ROJAS JAVIER GUADALUPE

REYES ALCARAZ LUIS ARMANDO

TAMEZ GUZMÁN RAFAEL

ASESOR DE TESIS
M. I. JUAN CARLOS ROA BEIZA

MÉXICO, D.F. 2009

ÍNDICE TEMÁTICO

CAPÍTULO 1

1.1 INTRODUCCIÓN

El siguiente trabajo describe el proceso que se siguió para el desarrollo de un sistema de control de información en una biblioteca dedicada a la recopilación y préstamo de libros, así como de material gráfico. Esta tarea comprende el manejo de usuarios en cantidad considerable por lo cual se requiere de un tratamiento de la información muy ágil en el proceso de su gestión. Son varias las bibliotecas importantes que han invertido en este tipo de automatización de la cual han obtenido un gran progreso además de la reducción en sus costos de operación.

En la literatura bibliotecológica existe una gran variedad de conceptos utilizados para definir un Integrated Library Systems (ILS, sistema integral para la automatización de las bibliotecas). Luis Ángel García Melero, escritor español, lo define como “un conjunto organizado de recursos humanos que utilizan dispositivos y programas informáticos, adecuados a la naturaleza de los datos que deben procesar, realizar y facilitar los servicios de la biblioteca, que son: almacenar de forma organizada el conocimiento humano contenido en todo tipo de materiales bibliográficos para satisfacer las necesidades informativas, recreativas y de investigación de los usuarios”[1]

A partir del nacimiento del formato MARC (Machine Readable Cataloguing Record) para el almacenamiento de registros bibliográficos, los sistemas de automatización de bibliotecas se consolidaron a finales de la década de los años 1970. En los albores de los años 1980 se establecieron las bases del concepto de sistema integrado. Estos sistemas para la automatización de bibliotecas surgieron como una evolución de los sistemas monofuncionales que se emplearon hasta finales de los años 1970, los cuales

[1] García Melero LA. Automatización de bibliotecas. Madrid: Arco/Libros. 1999.

tenían por objetivo resolver el problema de la gestión mecánica de funciones que suponían un mayor costo de recursos humanos a las grandes bibliotecas. A partir de la década de los años 1980, se comenzó a considerar el momento de los sistemas integrados, completos, centrados y únicos.[2]

La estructura bibliotecaria en México viene siendo una parcela abandonada en la incorporación de técnicas de gestión, pese a ser la base de la estructura en la que se sustentan tanto el proceso de modernización como otros procesos encaminados a facilitar la accesibilidad a la información. Partiendo del punto que la biblioteca es una unidad operacional total y todos sus diferentes operadores están interrelacionadas creemos necesario diseñar un sistema eficiente, único y de integración total en línea, que incluya todas las operaciones que se puedan manejar en ella para poder afrontar las cargas de trabajo crecientes día a día, asociado con personal capacitado para operarlo de manera efectiva, logrando el aprovechamiento máximo del sistema y desencadenando una eficiencia absoluta. La necesidad de ubicar el material de manera fácil requiere de un control de la disponibilidad de toda la colección; la cantidad de personas que acuden a biblioteca y la frecuencia con que lo hacen, requieren de un registro y organización.

La consulta debe realizarse mediante una aplicación con entorno gráfico que permita al bibliotecario bien capacitado ejecutar de manera automatizada todos los procesos implicados en la tarea bibliotecaria: préstamo de libros, altas y bajas de usuarios, monitoreo diario de movimientos y cobranzas, etc. de manera sencilla. El usuario debe ser pensado como alguien no especialista en informática y la consulta para éste podrá realizarse local o remotamente, mediante una interfaz amigable que deberá soportarse bajo un sistema operativo de fácil manejo. Podrá realizar consultas de libros en existencia sin la necesidad de llevar a cabo el llenado de formatos complejos que

[2] Jacquesson A. la información de las Bibliotecas: historia, estrategia y perspectivas. París: Circulo de la Biblioteca. 1995.

dilapidan tiempo, préstamos y movimientos que hayan hecho anteriormente, así como sus adeudos en una forma fácil y directa. De esta manera se emplearán todas las posibilidades que la tecnología puede brindar y de igual modo se logrará satisfacer totalmente las necesidades del usuario.

El objetivo es instaurar un sistema de almacenamiento y difusión de información haciendo de la biblioteca un “Sistema de Información” que presente numerosas ventajas, accesible desde cualquier computadora conectada a la red local de la institución de la que depende o desde cualquier punto de la red Internet. Todas las funciones deberán partir de una misma pantalla, evitando salidas y entradas en diferentes módulos que desorientan y retardan el trabajo, de esta manera obtendrá la agrupación de la información en un único lugar, estructurada y siempre accesible, permitiendo la actualización de la misma instantáneamente y aumentando la rapidez y fiabilidad de los procesos.

El sistema que proponemos será diseñado e implantado como una serie de módulos basado en la metodología UML permitiendo el tratamiento particular de cada tarea.

Destacando su economía se desarrollará sobre sistemas compatibles haciendo que la inversión no sea elevada, de igual manera se utilizará una base de datos gratuita; como entorno de desarrollo el lenguaje de programación Delphi para que pueda ser trasportada a cualquier plataforma. Se estructura en cuatro capítulos:

En la primera etapa se traza el problema actual en la institución bibliotecaria, su estructura operativa, la logística de su funcionamiento y el análisis de los requerimientos actuales que enfrenta.

Enseguida se aborda una exploración de la posible metodología a seguir, las características, ventajas y desventajas de las herramientas de desarrollo y bases de datos necesarias para elaborar el proyecto.

Posteriormente se describe la problemática actual de la institución, una recopilación y el análisis de información, la comparación de las herramientas de solución y la identificación de los posibles módulos del sistema.

Por último se tratará el diseño y la construcción del sistema propuesto, la justificación de la metodología, el bosquejo del trabajo, el diseño y construcción de la interfaz, la integración del trabajo y las pruebas aplicadas del sistema, así como la obtención de reportes.

1.2 ESTRUCTURA OPERATIVA DEL SISTEMA DE BIBLIOTECAS

Las bibliotecas, ya sean públicas, escolares o especializadas, tienen una organización interna que les permite brindar de manera adecuada los servicios que ofrecen. Pocas personas conocen cómo funciona en su interior una biblioteca, en el presente proyecto es de vital importancia conocer la organización interna de una biblioteca para poder satisfacer las necesidades de los usuarios de manera óptima.

Las bibliotecas se dividen en departamentos para cumplir con sus principales objetivos, los cuales son; informar, educar y recrear. Los departamentos mencionados son; administración, organización, dirección, servicios técnicos, consulta, préstamo interno y externo, libros de reserva, reprografía o fotocopiado y en algunos casos el departamento de audiovisuales.

Departamento de Administración

El responsable de la administración de una biblioteca es su director, quien se encarga de organizar los servicios y las actividades en la misma. Además diseña los programas para comprar y actualizar el material de lectura.

La Dirección

El director o directora es la máxima autoridad en una biblioteca, es quien establece el reglamento para cada uno de los servicios que se brindan y las sanciones para aquellos que no los cumplan.

Organización

La organización de los materiales adquiridos por la biblioteca corresponde a este departamento en el que trabajan bibliotecarios, personas con estudios universitarios especializadas en organizar la información. De esta área depende que los usuarios puedan encontrar de manera fácil la información que necesiten.

Servicios Técnicos

El departamento de servicios técnicos es el responsable de seleccionar el material que la biblioteca debe comprar para brindar un servicio adecuado, según el tipo de usuarios para la que esté diseñada.

En él también se preparan los libros que se pondrán en circulación, continuamente se revisan los ejemplares para garantizar que se encuentren en buenas condiciones, se les asigna un número de adquisición, se les pega el sobre para poner la tarjeta de préstamo, se sellan con el nombre de la biblioteca y se elaboran sus respectivas tarjetas catalográficas que registran los datos bibliográficos, tales como autor, título, traductor, editorial, número de páginas, etcétera.

Área de Consulta

El área de consulta es una de las más importantes en la organización de la biblioteca. Está a cargo de un bibliotecario, quien debe conocer los títulos de los libros con los que cuenta y su ubicación física.

Al área de consulta acuden usuarios para solicitar información, para ubicar los libros y todo tipo de material que ofrece la biblioteca. Esta área también responde las solicitudes telefónicas y por correo electrónico de los usuarios.

Departamento de Préstamo Interno y Externo

Las personas que laboran en el departamento hacen el registro de los lectores que están autorizados para realizar consultas externas de la biblioteca en calidad de préstamo; también son los responsables de elaborar las credenciales de préstamo a domicilio y de solicitar las identificaciones de los usuarios internos. Asimismo, controlan la existencia de los libros que están disponibles tanto para consulta interna como externa.

Algunas bibliotecas cuentan con el servicio de préstamo inter–bibliotecario, lo cual permite al usuario obtener en préstamo los textos de otras bibliotecas, por medio de los convenios que la biblioteca tenga con otras.

Área de Reserva

El área de reserva es la responsable de mantener los libros más demandados por los usuarios, disponibles en todo momento y cuida que esos ejemplares se mantengan en las mejores condiciones posibles, por lo que no se prestan a domicilio.

Departamento de Reprografía o Fotocopiado

En el departamento de reprografía o fotocopiado están las personas encargadas de fotocopiar los textos que el usuario necesita, ellos indican el número máximo de fotocopias que se pueden obtener y si está permitido o no reproducir un libro en particular.

Audiovisual

Generalmente es un salón aparte de la sala de consultas, para no molestar a los demás usuarios, donde pueden verse videos, filminas, diapositivas, discos compactos y escuchar grabaciones.

Petición de libros

Un usuario puede realizar una petición de uno o más libros a la biblioteca. Presenta el carnet de usuario de la biblioteca y una ficha en la que se detallan los libros pedidos. Una vez entregados el carnet y la ficha, el personal comprobará y aceptará la petición de los libros solicitados siempre que pueda satisfacer la petición, es decir, cuando haya ejemplares disponibles. Si se acepta la petición, se actualiza el número de unidades de los libros de la biblioteca y se guarda la ficha de préstamo. Figura 1.2.1 y 1.2.2.

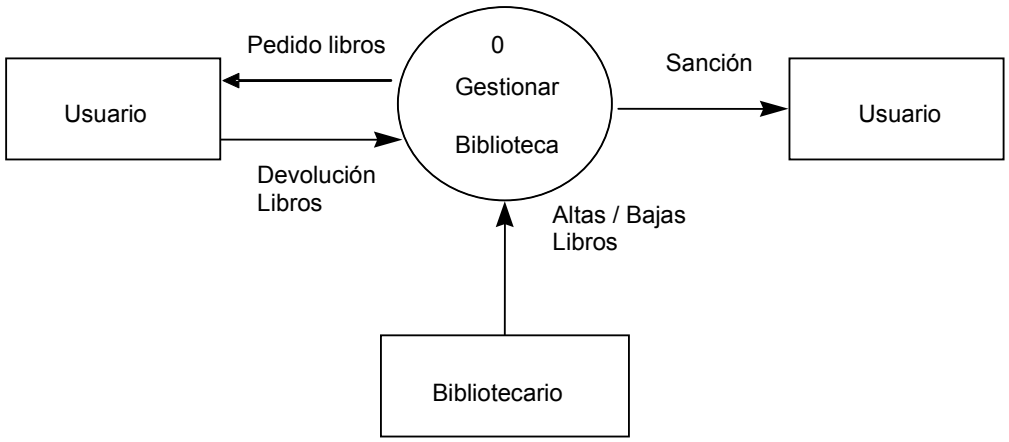


Figura 1.2.1 Diagrama de contexto

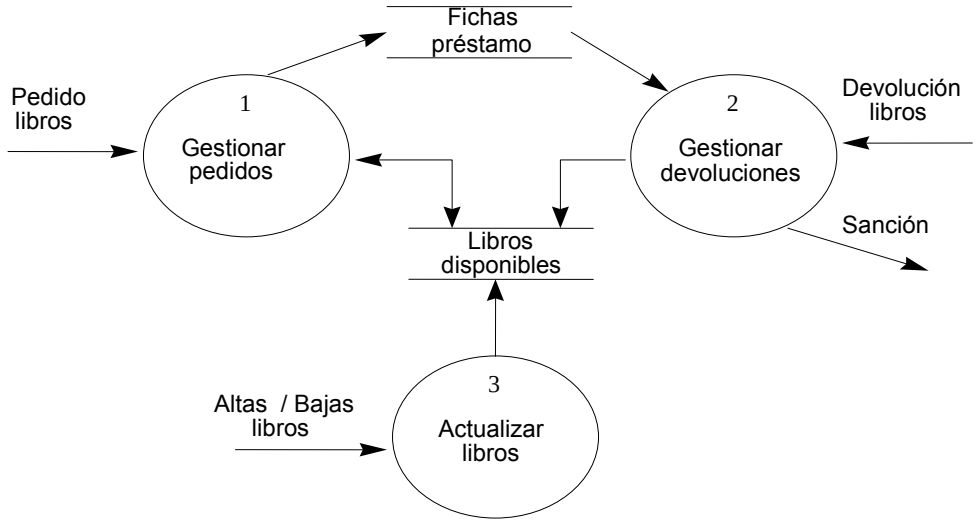


Figura 1.2.2 Gestionar biblioteca

Devoluciones de libros

Un usuario no puede solicitar más préstamos hasta que no haya efectuado todas las devoluciones del préstamo anterior. Para realizar un préstamo necesita el carnet, que no le es entregado hasta que no haya devuelto todos los libros solicitados, sin embargo es posible hacer una devolución parcial de los ejemplares. Cuando un usuario realice una devolución, el personal actualizará el stock de libros y comprobará la fecha de devolución de cada ejemplar para estudiar, en el caso de que la devolución se haga fuera de tiempo, la imposición de una sanción que tiene un costo de X unidades monetarias por cada ejemplar y días de retraso en la devolución. En este caso, la sanción se emite cuando el usuario entrega el último ejemplar. El bibliotecario se encarga de las altas y bajas de los libros de la biblioteca. Ver figura 1.2.3

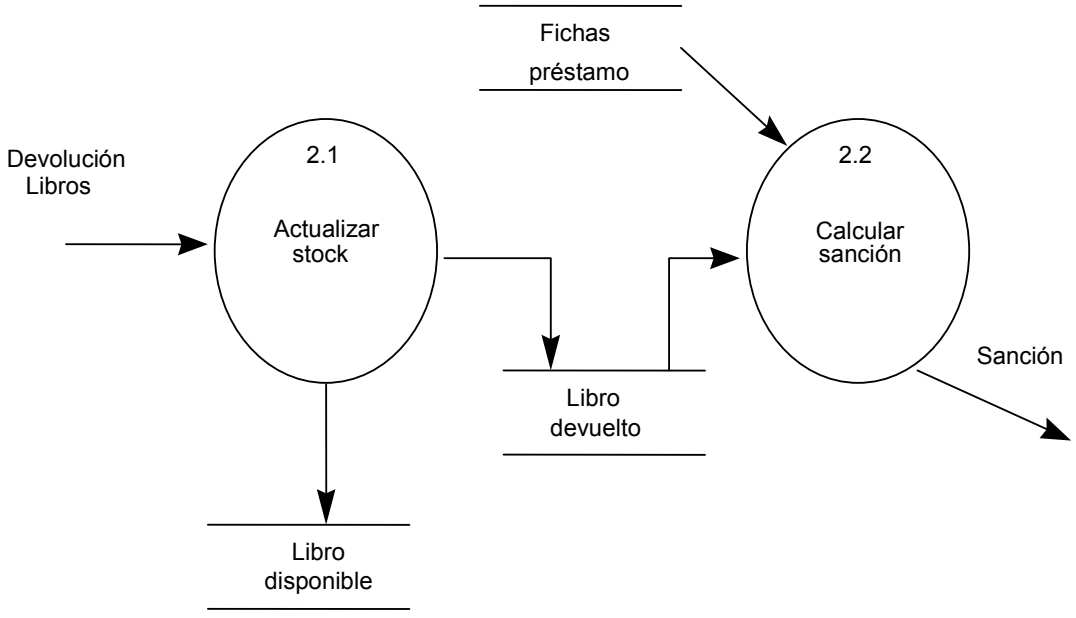


Figura 1.2.3 Gestionar devoluciones

En la figura 1.2.4 se muestra, usando UML, en un sólo diagrama la operación del sistema de bibliotecas descrita anteriormente.

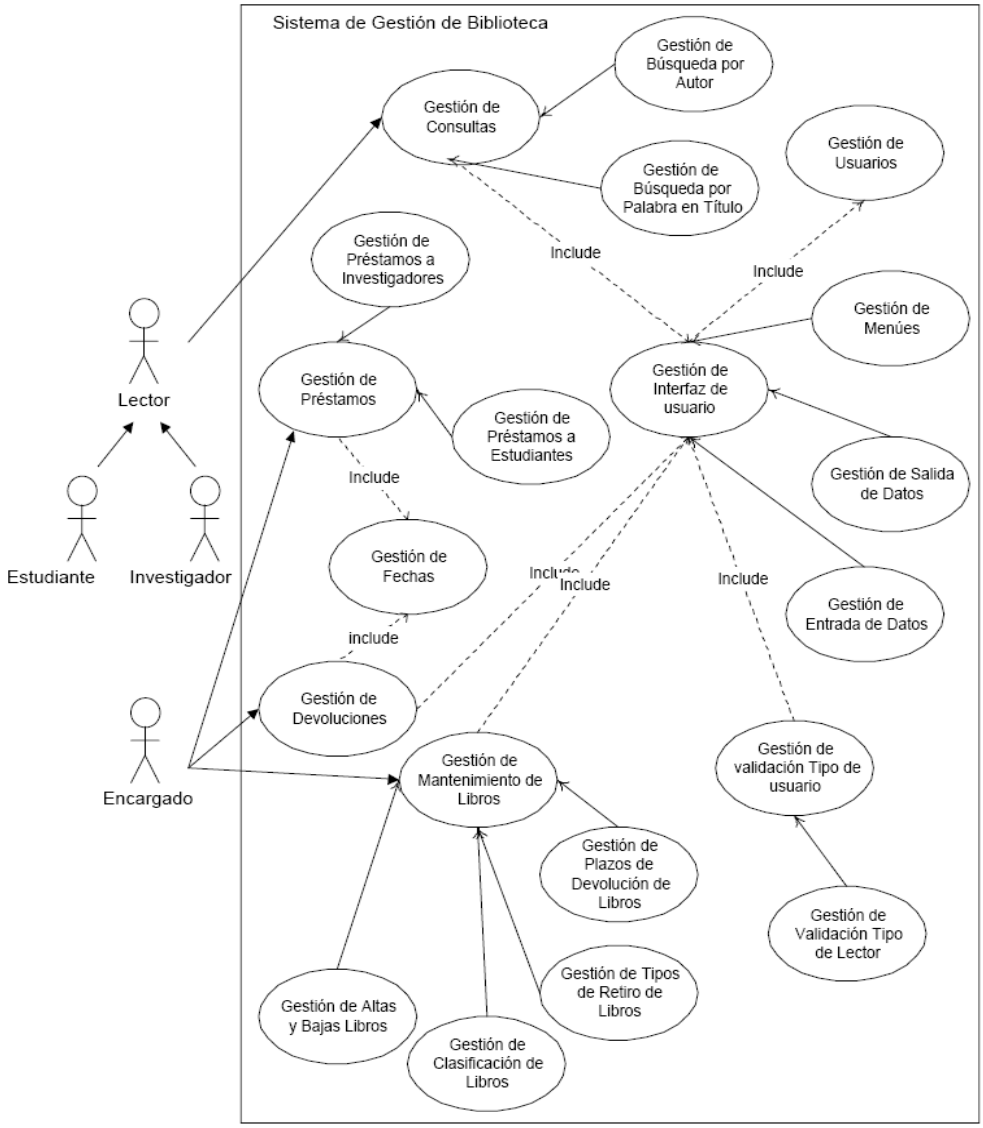


Figura 1.2.4 Diagrama de operación

1.3 LOGÍSTICA OPERATIVA DEL FUNCIONAMIENTO DEL SISTEMA DE BIBLIOTECAS

En el presente capítulo se describirá la actual forma en la cual los usuarios de la biblioteca realizan su registro personal, registro para préstamo externo, manejo de sanciones en los casos en los que los usuarios no devuelvan el libro en el tiempo estimado, así como la clasificación que el personal de la biblioteca realiza de manera manual para catalogar los ejemplares.

Es preciso crear o adoptar un sistema de clasificación, “clasificar” es la operación mediante el cual se establece qué clase o clases se debe adscribir un libro o documento, a partir del análisis de los documentos de que se compone su contenido, y de acuerdo con un sistema de clasificación determinado.

Una clasificación bibliográfica es un conjunto de términos y procedimientos utilizados para representar un contenido, pueden ser de estructura jerárquica, para lo cual se tendrá que emplear notaciones convencionalmente codificadas que permitan una ordenación jerárquica; o bien puede ser de estructura asociativa, en cuyo caso se emplearán palabras clave o términos asociados.

Los lenguajes documentales de estructura jerárquica se pueden dividir en clasificaciones enciclopédicas, algunas de ellas son, Bliss, Dewey, Clasificación Decimal Universal **(CDU)**. En los lenguajes de estructura asociativa basta con ordenar alfabéticamente las palabras o grupos de palabras que expresan los conceptos fijados en el proceso de clasificación

Los bibliotecarios han catalogado la biblioteca siguiendo la Clasificación Decimal Universal **(CDU)**, el sistema CDU fue publicado por primera vez en 1904 y 1907, desde su publicación y hasta la actualidad ha sido continuamente revisado, por lo que ha

demostrado ser un sistema flexible para clasificar fondos bibliográficos. La biblioteca se encuentra trabajando con este sistema en su última edición que data del año 2004.

En un principio la CDU dividió el conocimiento en diez clases principales. Posteriormente se agruparon en nueve dejando la clase 4, esta clase había estado ocupada hasta entonces por la filología. Con las clases restantes se crea la tabla principal (tabla 1.3.1), cada campo del conocimiento a su vez se divide en otros diez y así sucesivamente sin límite.

Grupo principal	Descripción	Ejemplos de subniveles
0	Generalidades.	Ciencia y conocimiento.
	Organización.	Información.
	Documentación.	Enciclopedias
1	Filosofía	Psicología
2	Religión	Teología.
3	Ciencias sociales.	Estadística.
		Política.
		Economía.
		Comercio.
		Derecho.
		Gobierno.
		Asuntos militares.
		Bienestar social.
		Seguros.
		Educación.
		Folklore.
4	(Vacante)	-----
5	Matemáticas	Ciencias naturales.
6	Ciencias aplicadas.	Medicina.
		Tecnología.

7	Bellas artes.	Juegos. Espectáculos. Deportes.
8	Lenguaje.	Lingüística. Literatura.
9	Geografía	Biografías. Historia.

Tabla 1.3.1 Clasificación Decimal Universal (CDU)

La clasificación basa su ordenación del conocimiento a través de dígitos, a un grupo principal se le asigna un dígito del 0 al 9 y a cada nivel que sea creado dentro del grupo se le va añadiendo un nuevo dígito. Por ejemplo:

3 – Ciencias Sociales

34 – Derecho

341 – Derecho Internacional.

Los usuarios de los servicios bibliotecarios deben de buscar el libro que desean consultar en fichas según la clasificación CDU, en la ficha (Tabla 1.3.2) se muestran reflejados los datos, además de las claves para saber la ubicación exacta en la biblioteca.

CDU 15	**Materia: Psicología**
Autor:	**BLÁ**ZQUEZ. M.A
Titulo:	**Cla**ves de la psicología
Editorial, lugar, año:	Salvat, Barcelona, 1986
Observ:	Descripción física: 20 cm., 64 p. Il.
	No. Registro: 2283
	ISBN 84 345 7860 3

Tabla 1.3.2 Fichas utilizadas por los usuarios para localizar el material

Una vez que los usuarios localizan el libro deseado en los ficheros lo buscan personalmente en la biblioteca, los libros igualmente están clasificados con una marca en el lomo llamada tejuelo (figura 1.3.3), para una mejor ubicación.

No. CDU	**15**	(Psicología)
3 primeras letras del autor	**BLA**	(Manuel Blázquez)
3 primeras letras del título	**CLA**	(Claves de Psicología)

Figura 1.3.3 Tejuelo

Este tejuelo pertenece al libro “Claves de la Psicología” el autor es Manuel Blázquez, por lo que se encuentra en la estantería donde se colocan los libros clasificados con el número 1 que corresponde al grupo general de Filosofía. Como la Psicología es una subdivisión de la Filosofía y en el CDU se le otorga el número 15.

Todos los libros de la misma materia se ordenan alfabéticamente por el apellido del autor, los ejemplares de un mismo autor se clasifican alfabéticamente por sus títulos.

Dentro de la biblioteca los usuarios pueden hacer uso de los recursos bibliográficos, pero si algún usuario desea préstamo a domicilio, entonces deberá llenar manualmente otros datos como se indican en la ficha (figura 1.3.4) además de proporcionar dos referencias.

	Número de usuario: _____
Nombre del usuario:	_____
Dirección:	_____
No. de cuenta:	_____
Teléfono:	_____
Teléfono celular:	_____
Correo electrónico:	_____
Escuela:	_____
Primera referencia	
Nombre:	_____
Dirección:	_____
Teléfono:	_____
Segunda referencia	
Nombre:	_____
Dirección:	_____
Teléfono:	_____
Atendido por: _____	Firma: _____

Figura 1.3.4 Ficha para el alta de usuarios

Los recursos bibliográficos son prestados por 4 días hábiles, solamente se pueden prestar dos recursos por usuario, siempre y cuando el recurso esté disponible y no se encuentre dentro de los ejemplares exclusivos para consulta interna. Los diccionarios, enciclopedias revistas y periódicos no se prestan, la renovación del préstamo se realiza por el mismo plazo 4 días hábiles, siempre y cuando no se haya sobrepasado la fecha de devolución.

Una vez que los usuarios han llenado su ficha y están dados de alta entonces deberán llenar dos papeletas por material bibliográfico que deseen extraer de la biblioteca. La

figura 1.3.5 muestra la papeleta que los usuarios deben de completar, una es para el usuario y otra para el personal de la biblioteca para tener un control del material prestado así como las fechas de devolución.

Ficha para préstamo domiciliario			
Fecha:	_____	Hora:	_____
No. Registro:	_____		
Autor:	_____		
Titulo:	_____		
Datos del Usuario			
Numero de Usuario:	_____		
Nombre del usuario:	_____		
Dirección:	_____		
No. de cuenta:	_____		
Teléfono:	_____		
Solo para el personal de la biblioteca:			
Atendido por:	_____	Firma:	_____
Fecha de préstamo:	_____	Fecha de devolución:	_____

Figura 1.3.5 Papeleta para préstamo domiciliario

Los usuarios que no entreguen el material en la fecha de devolución se hacen acreedores a una multa, la cual consiste en el pago simbólico en efectivo, así como la suspensión del servicio de préstamo domiciliario por un periodo de 4 días hábiles; si el usuario reincide por segunda vez, entonces la suspensión será por 8 días hábiles, si

reincide por tercera vez, su servicio será dado de baja definitivamente. Las notas de reincidencia de los usuarios son anotadas en la ficha de alta de usuarios y cada vez que un usuario desea un préstamo domiciliario el personal de la biblioteca debe consultar la ficha para conocer el estatus del usuario.

1.4 ANÁLISIS DE LA PROBLEMÁTICA DEL SISTEMA DE BIBLIOTECAS

Introducción

La biblioteca es considerada un servicio que contribuye al desarrollo de los derechos humanos, en especial al derecho de la libertad de pensamiento, información y opinión. Estos recintos se encuentran sometidos a una constante y profunda evolución, por lo cual vale la pena realizar una introspección hacia la problemática que un sistema de bibliotecas representa.

Uno de los principales problemas de una biblioteca es la localización rápida y confiable del material que resguarda, para esto se hace necesaria la introducción y uso apropiado del equipo de cómputo así como de diversos productos informáticos que faciliten la gestión de la información, o al menos de ciertas facetas de ella, todo esto reunido en un programa de gestión que brinde una alta capacidad de procesamiento, velocidad, flexibilidad y precisión.

En los años 80 una tendencia en la automatización de las bibliotecas llevó a la aparición de los llamados sistemas integrados de gestión de bibliotecas (SIGB) y los catálogos en línea de acceso público (OPAC). Su implantación benefició la integración y la colaboración de las diferentes áreas en el interior de la organización.

Más tarde con el surgimiento de Internet y el acceso, cada vez más común, por parte de las personas a las computadoras, condicionó que estos productos experimentaran adaptaciones para buscar trascender las fronteras institucionales, facilitar la comunicación y el servicio a los usuarios.

La problemática sigue vigente por lo cual surge la necesidad de desarrollar una aplicación SIGB que se ajuste a las necesidades y características generales propias de

una biblioteca que permita el acceso rápido y confiable a la información contenida en la misma.

La Biblioteca como sistema y la Automatización

En la Sociedad del Conocimiento se observa una gran oferta y demanda social de información. La Biblioteca ha de adaptarse a ello y actuar como intermediaria del conocimiento, pensamiento y cultura.

Por ello (unido a otros motivos económicos), se ha visto la necesidad de que la biblioteca se entienda como un sistema, que se define como un conjunto de elementos en interacción dinámica para la consecución de unos objetivos en un entorno.

Así la automatización se convierte en pieza clave para que la biblioteca como sistema (cada vez más complejo en sus servicios y funcionamiento) consiga estos objetivos. Se tiende a aplicar hoy al sistema total de la biblioteca, para abarcar todas las tareas bibliotecarias de forma integrada en un único sistema automatizado.

La automatización concepto y evolución hasta llegar a los Sistemas Automatizados

Se podría definir la automatización en bibliotecas como la aplicación de herramientas informáticas en las tareas bibliotecarias. De usar tareas documentales puntuales mediante pequeñas aplicaciones pasó a aplicarse actualmente un sistema total de bibliotecas mediante el SIGB.

La automatización ha sido un proceso paulatino en los cuales pueden puntualizarse los siguientes aspectos:

- Comenzó en los años sesenta.
- En los años setenta, con la aparición de las microcomputadoras y su abaratamiento, la automatización se comenzó a aplicar a los catálogos y

posteriormente se pensó en la cooperación bibliotecaria favorecida por reciente nacimiento de las telecomunicaciones, además de nuevas normas y formatos los cuales unificaban criterios.

- En los años ochenta nacen los catálogos de acceso público en línea, OPAC **(Online Public Access Catalog)** y los primeros SIGB's, como Dobis/Libis, Aleph y Sabini.
- Desde los años noventa a la actualidad se va alcanzado una madurez en esta disciplina. Aparecen los SGBD para el control de la colección y posteriormente se desarrollan los grandes SIAB (Sistema Integrado de Automatización de Bibliotecas), que amplían la automatización a todas las tareas bibliotecarias, integrando todas las funciones en un único sistema automatizado.
- Con la llegada de Internet se desarrollan grandes redes cooperativas, permitiendo compartir recursos y ofrecer servicios en línea. Se amplían las funciones de estos sistemas en este sentido; hoy en día algunos autores los han denominado SIGB Extendidos.

El por qué de la automatización

La automatización en una biblioteca se debe fundamentalmente a que el usuario demanda mayores prestaciones por lo tanto la biblioteca debe satisfacerlas y la automatización ofrece herramientas para hacerlo.

Así pues la automatización puede justificarse por:

- El colapso del sistema manual ante el gran volumen de información.
- Por la necesidad de agilizar procesos y optimizar recursos.
- Para detectar las carencias y subsanarlas, revisando y reorganizando los procedimientos.
- Para disponer de información actualizada del funcionamiento de la biblioteca (como las estadísticas) que permitan una evaluación y mejoras.

- Y para facilitar la cooperación bibliotecaria, compartiendo recursos permitiendo ofrecer mejores servicios con un costo menor.

Los sistemas integrados de automatización de bibliotecas

Al entender la biblioteca como un sistema, y al tratar de mantener su eficiencia en la automatización, se ha pasado del uso de aplicaciones informáticas individuales al uso de sistemas integrados. Estos sistemas se integran en un sólo programa, distintas aplicaciones específicas para cada tarea (llamadas módulos) que están interrelacionadas entre sí y comparten las mismas bases de datos, con esto se evita la redundancia de información y aumenta su eficacia.

Un sistema integrado modular permite gestionar todas las funciones y servicios de la biblioteca de forma automatizada. Los módulos de un SIGB se gestionan de forma independiente, pero todos ellos se afectan unos a otros en términos de información. De ahí que se les denomine: Sistemas Integrados de Gestión Bibliotecaria (SIGB).

Características de los SIGB

En la actualidad existen una oferta variada de éstos sistemas en el marcado. Por lo regular son desarrollados por empresas privadas que intentan adaptarse a las necesidades bibliotecarias.

Entre las características más comunes se encuentran las siguientes:

- Trabajan con una arquitectura cliente-servidor.
- Se ajustan a normas y estándares oficiales y de mercado.
- Comparten la misma información de una BD.
- Suelen ser sistemas abiertos, ya que pueden ejecutarse en diferentes sistemas operativos.

Módulos y Funciones

Los SIGB contienen los siguientes módulos para tareas bibliotecarias específicas:

- Módulo de Selección y Adquisición. Se encarga de gestionar la selección (con tareas de información bibliográfica sobre ejemplares, editores, etc.) y la adquisición (mediante la compra, canje o donación) llevando a cabo los pedidos y la gestión económica y presupuestaria que conlleva.
- Módulo de Catalogación. Permite gestionar el catálogo, incluyendo en algunos, el control de autoridades.
- Módulo del Control de Autoridades. En algunos sistemas se integra dentro del módulo de catalogación. Permite validar y normalizar los puntos de acceso del catálogo.
- El Módulo de Recuperación de Información (OPAC – Catálogo público de acceso en línea). Permite recuperar información bibliográfica (por diferentes puntos de acceso) y ofrece información no bibliográfica, como disponibilidad de ejemplares, reservas, etc. Ha evolucionado notablemente en su diseño, prestaciones y accesos, planteándose hoy un nuevo concepto de OPAC en soporte web, que manteniendo los atributos del OPAC tradicional, permite el acceso en línea, soporta hipertexto, recupera recursos digitales, etc. En este sentido, las últimas tendencias tienden a extender sus funciones, ofreciendo más información y servicios integrados en único espacio web.
- Módulo de Circulación. Gestiona el préstamo, devolución, renovación, etc. Contempla los distintos tipos de préstamos y especialmente ha sido de gran ayuda para la gestión y control del préstamo interbibliotecario debido a su mayor complejidad. Además permite realizar: credenciales de usuarios, cartas de reclamación, avisos, reservas, etc.
- Módulo de Publicaciones Periódicas. Permite su control y adquisición, pero por sus características de mayor complejidad, este módulo no se ha desarrollado mucho todavía. El control de las publicaciones periódicas plantea bastantes problemas porque son extremadamente dinámicas, especialmente las

electrónicas que además no se adquieren físicamente sino que se compran licencias de uso, dificultando su control. Por ello los sistemas no ofrecen aún una buena funcionalidad y sencillez en este módulo, y ofrecen escasa integración con el vaciado de revistas y con la publicación a texto completo.

- Módulo de Estadísticas. Produce estadísticas sobre los procesos para establecer indicadores que ayuden a la gestión y a la toma de decisiones.
- Módulo de Administración y Gestión. Proporciona ayuda para la organización y gestión de tareas.

Por otro lado se tienen los SGBD (Sistemas de Gestión de Bases de Datos) que aunque no son SIGB, su funcionalidad se ha adaptado y se utilizan en ellos; los SGBD son herramientas para organizar la información de una forma eficaz, pueden ser clasificados en dos grandes rubros:

- Relacionales: Útiles para tratar información muy estructurada como los registros catalográficos.
- Documentales: Útiles para organizar información no estructurada como resúmenes, documentos a texto completo, etc.

Últimas Tendencias de Sistemas Automatizados de Bibliotecas

Con la aparición de Internet y su aplicación en los servicios bibliotecarios se ha conformando un nuevo concepto de biblioteca. Algunos autores difieren en su definición sin embargo se pueden encontrar como biblioteca Digital, Híbrida, Electrónica, Virtual, etc. Estos sistemas son definidos como SIGB Extendido, que amplía sus funciones y servicios a Internet haciendo uso de los siguientes módulos:

- Gestión conocimiento corporativo.
- Gestión documental de la colección.
- Gestión de servicios y productos informativos.
- Gestión de portales web de servicio.

1.5 REQUERIMIENTOS DEL SISTEMA DE BIBLIOTECAS, SOFTWARE Y HARDWARE

Los requerimientos mínimos para la implementación con relación al software son:

Requerimientos en Servidor:

- Microsoft Windows 2000, XP.
- Base de datos Firebird en su versión 1.5
- Instalación de aplicación de biblioteca.
- Instalación de librerías dinámicas en los módulos de instalación del aplicativo del Sistema de Bibliotecas.

Requerimientos en Cliente:

- Instalación del aplicativo cliente del Sistema de Bibliotecas.
- Instalación de librerías dinámicas DLL.

Con relación al hardware mínimo en servidor es:

- Intel Pentium II/233 MHz o superior.
- 512 MB RAM (1 GB recomendado).
- 75 MB de espacio en disco duro cliente.
- 160 MB de espacio en disco duro para instalación completa.
- SVGA o superior en resolución.

CAPÍTULO 2

2.1 BASE DE DATOS RELACIONALES, METODOLOGÍA UML

Definición de base de datos

Una base de datos se define como una serie de datos organizados y relacionados entre sí. Dichos datos son recolectados y explotados por el sistema de información de la empresa o negocio.

Características

Entre las principales características podemos mencionar:

- Independencia lógica y física de los datos.
- Redundancia mínima.
- Acceso concurrente por parte de múltiples usuarios.
- Integridad de los datos.
- Consultas complejas optimizadas.
- Seguridad de acceso y auditoria.
- Respaldo y recuperación.
- Acceso a través de lenguajes de programación estándar.

Sistema de Gestión de Base de Datos (SGBD)

Los Sistemas de Gestión de Base de Datos (en inglés DataBase Management System) son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta.

Modelo entidad-relación

Los diagramas o modelos entidad-relación (denominado por su siglas, ERD “Entity Relationship Diagram”) son una herramienta para el modelado de datos de un sistema de información. Estos modelos expresan entidades relevantes para un sistema de información, sus interrelaciones y propiedades, figura 2.1.1.

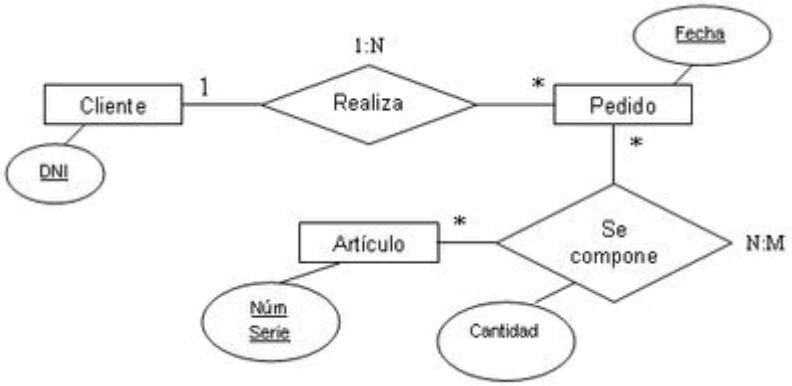


Figura 2.1.1 Sistema de información, sus interrelaciones y propiedades

Cardinalidad de las Relaciones

El diseño de relaciones entre las tablas de una base de datos puede ser la siguiente:

- **Relaciones de uno a uno:** una instancia de la entidad A se relaciona con una y solamente una de la entidad B.
- **Relaciones de uno a muchos:** cada instancia de la entidad A se relaciona con varias instancias de la entidad B.
- **Relaciones de muchos a muchos:** cualquier instancia de la entidad A se relaciona con cualquier instancia de la entidad B.

Álgebra Relacional

Es un conjunto de operaciones que describen paso a paso cómo computar una respuesta sobre las relaciones, tal y como éstas son definidas en el modelo relacional. Describe el aspecto de la manipulación de datos. Estas operaciones se usan como una representación intermedia de una consulta a una base de datos y, debido a sus propiedades algebráicas, sirven para obtener una versión optimizada y eficiente de dicha consulta.

Tuplas

Son las "filas de una tabla" en el lenguaje usual de bases de datos.

Unión compatible

Una unión es compatible entre dos relaciones, si ellas poseen el mismo grado y su dominio son los mismos de izquierda a derecha.

Grado

Número de atributos.

Operaciones

Selección (σ)

Una condición puede ser una combinación booleana, donde se pueden usar operadores como: $\wedge, \vee$, combinándolos con operadores $<, >, \leq, \geq, =, \neq$.

Proyección (Π)

Permite extraer columnas (atributos) de una relación, dando como resultado un subconjunto vertical de atributos de la relación, esto es:

$\Pi_{A_1,A_2,...,A_n}(R)$, donde $A_1, A_2, \ldots, A_n$son atributos de la relación R.

Producto cartesiano (x)

El producto cartesiano de dos relaciones se escribe como:

$R \times S$

Entrega una relación cuyo esquema corresponde a una combinación de todas las tuplas de R con cada una de las tuplas de S, y sus atributos corresponden a los de R seguidos por los de S.

Unión (∪)

Se denota como:

$R \cup S$

Regresa el conjunto de tuplas que se encuentran en R, o en S, o en ambas. R y S deben ser uniones compatibles.

Diferencia (-)

La diferencia de dos relaciones, R y S denotada por:

$R - S$

Entrega todas aquellas tuplas que están en R, pero **no** en S. R y S deben ser uniones compatibles.

Estas operaciones son fundamentales en el sentido en que:

- Todas las demás operaciones pueden ser expresadas como una combinación de éstas
- Ninguna de estas operaciones pueden ser omitidas sin que con ello se pierda información.

Cálculo Relacional

Es un lenguaje de consulta que describe la respuesta deseada sobre una Base de datos sin especificar cómo obtenerla, a diferencia del Álgebra relacional que es de tipo declarativo.

UML (Unified Modeling Languaje)

UML significa "Unified Modeling Language": Lenguaje de Modelado Unificado. Es un lenguaje usado para especificar, visualizar y documentar los diferentes aspectos relativos a un sistema de software en desarrollo, así como para modelado de negocios y otros sistemas no necesariamente de software.

Puede ser utilizado con cualquier metodología a lo largo del proceso de desarrollo de software, en cualquier plataforma tecnológica de implementación (Unix, Windows etc.).
Es un sistema notacional destinado a los sistemas de modelado que utilizan conceptos orientados a objetos. Los principales factores que motivaron la definición de UML fueron la necesidad de modelar sistemas, las tendencias en la industria del software, la unificación de distintos lenguajes y métodos existentes además de innovación en los modelos para adaptarse a la arquitectura distribuida.

Es importante resaltar que un modelo UML describe lo que supuestamente hará un sistema, pero no dice cómo implementar dicho sistema. Para la construcción de modelos, hay que centrarse en los detalles relevantes mientras se ignoran los demás, por lo cual con un único modelo no se tiene lo suficiente. Por lo tanto varios modelos pueden aportar diferentes vistas de un sistema los cuales ayudan a comprenderlo desde varios frentes. Así, UML recomienda la utilización de nueve diagramas para representar las distintas vistas de un sistema.

Diagramas de UML

Diagrama de Casos de Uso: Modela la funcionalidad del sistema agrupándola en descripciones de acciones ejecutadas por un sistema para obtener un resultado. El diagrama es usado para entender el uso del sistema.
El diagrama de casos de uso muestra un conjunto de casos y actores (un actor puede ser tanto un sistema como una persona) y sus relaciones, es decir, muestra quién

puede hacer qué y las relaciones que existen entre acciones (casos de uso). Este diagrama es muy importante para moderlar y organizar el comportamiento del sistema.

Diagrama de Clases: Muestra las clases (descripciones de objetos que comparten características comunes) que componen el sistema y cómo se relacionan entre sí.

Diagrama de Objetos: Muestra una serie de objetos (instancias de las clases) y sus relaciones. A diferencia de los diagramas anteriores, este diagrama se enfoca en la perspectiva de casos reales o prototipos. Es un diagrama de instancias de las clases mostradas en el diagrama de clases.

Diagrama de Secuencia: Enfatiza la interacción entre los objetos y los mensajes que intercambian entre sí junto con el orden temporal de los mismos.

Diagrama de Colaboración: Igualmente, muestra la interacción entre los objetos resaltando la organización estructural de los objetos en lugar del orden de los mensajes intercambiados.
El diagrama de secuencia y el diagrama de colaboración muestran a los diferentes objetos y las relaciones que pueden tener entre ellos, los mensajes que se envían entre ellos. Son dos diagramas diferentes, que se pueden pasar de uno a otro sin pérdida de información, pero dan puntos de vista diferentes del sistema. En resumen, cualquiera de los dos es un Diagrama de Interacción.

Diagrama de Estados: Se utiliza para analizar los cambios de estado de los objetos. Muestra los estados, eventos, transiciones y actividades de los diferentes objetos. Son útiles en sistemas que reaccionen a eventos.

Diagrama de Actividades: Es un caso especial del diagrama de estados, simplifica el diagrama de estados modelando el comportamiento mediante flujos de actividades.

Muestra el flujo entre los objetos. Se utilizan para modelar el funcionamiento del sistema y el flujo de control entre objetos.

Diagrama de Componentes: Muestra la organización y las dependencias entre un conjunto de componentes. Se usan para agrupar clases en componentes o módulos.

Diagrama de Despliegue (o implementación): Muestra los dispositivos que se encuentran en un sistema y su distribución en el mismo. Se utiliza para identificar **sistemas de cooperación:** Durante el proceso de desarrollo el equipo averiguará de qué sistemas dependerá el nuevo sistema y que otros sistemas dependerán de él.

Clasificación de Diagramas

Se dispone de dos tipos diferentes de diagramas los que dan una vista estática del sistema y los que dan una visión dinámica. Figura 2.1.2

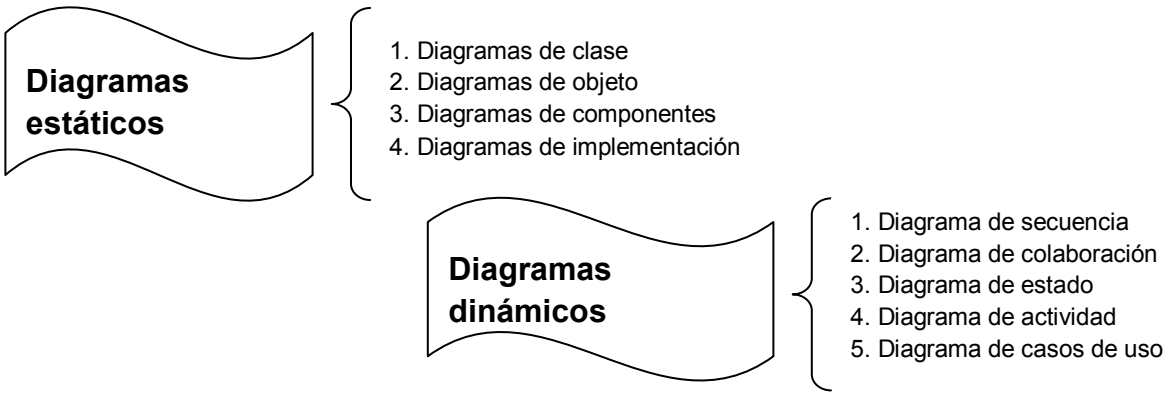


Figura 2.1.2 Clasificación de diagramas

La práctica de crear diagramas para visualizar sistemas desde perspectivas diferentes no está limitada a la industria de la construcción. En el contexto del software, existen cinco vistas complementarias que son las más importantes para visualizar, especificar, construir y documentar la arquitectura del software. En el UML las vistas existentes son:

1. Vista casos de uso: se forma con los diagramas de casos de uso, colaboración, estados y actividades.
2. Vista de diseño: se forma con los diagramas de clases, objetos, colaboración, estados y actividades.
3. Vista de procesos: se forma con los diagramas de la vista de diseño. Recalcando las clases y objetos referentes a procesos.
4. Vista de implementación: se forma con los diagramas de componentes, colaboración, estados y actividades.
5. Vista de despliegue: se forma con los diagramas de despliegue, interacción, estados y actividades.

Como es posible ver el número de diagramas es muy extensivo y en la mayoría de los casos podría decirse que son excesivos, sin embargo UML permite definir solamente los necesarios pues no todos los diagramas son necesarios para todos los proyectos. UML está pensando para el modelado de sistemas pequeños y sistemas complejos.

No se debe olvidar el protagonismo excesivo que se le da al diagrama de clases, el cual representa una parte importante del sistema, pero sólo representa una vista estática, o sea, muestra el sistema “parado”, por lo tanto con UML es posible saber su estructura pero no se puede saber lo que sucede en sus diferentes partes cuando el sistema empieza a funcionar.

2.2 CARACTERÍSTICAS, VENTAJAS Y DESVENTAJAS DE DELPHI

Delphi es un entorno de desarrollo de software diseñado para la programación de propósito general con énfasis en la programación visual. En Delphi se utiliza como lenguaje de programación una versión moderna de Pascal llamada Object Pascal.

Características

Una de las características más habituales de Delphi es el desarrollo de aplicaciones visuales y de bases de datos cliente-servidor y multicapas. Debido a que es una herramienta de propósito múltiple, se usa también para proyectos de casi cualquier tipo, incluyendo aplicaciones de consola, aplicaciones de web (por ejemplo servicios Web, CGI, ISAPI, NSAPI, módulos para Apache), servicios COM y DCOM, y servicios del sistema operativo.

Delphi inicialmente sólo producía ejecutables binarios para Windows: Delphi 1 para Win16 y con Delphi 2 se introdujo Win32. En la actualidad da más posibilidades:

- En la última versión RAD Studio 2009 incluye en el mismo entorno de desarrollo los lenguajes:
    - Delphi para Win32
    - Delphi para .NET
    - C# para .NET
    - C++
- Existe una versión de Delphi para sistemas Unix y Linux, denominada Kylix (de la cual existe un versión gratuita, aunque limitada). Sin embargo Kylix fue congelado por Borland en su versión 3.00.

Características del Lenguaje:

- Soporte para la programación orientada a objetos también existente desde Turbo Pascal 5.5, pero más evolucionada en cuanto a:
    - Encapsulación: declarando partes privadas, protegidas, públicas y publicadas de las clases
    - Propiedades: concepto nuevo que luego han adaptado muchos otros lenguajes. Las propiedades permiten usar la sintaxis de asignación para setters y getters.
    - Simplificación de la sintaxis de referencias a clases y punteros.
- Soporte para manejo estructurado de excepciones, mejorando sensiblemente el control de errores de usuario y del sistema.
- Programación activada por eventos (event-driven), posible gracias a la técnica de delegación de eventos. Esta técnica permite asignar el método de un objeto para responder a un evento lanzado sobre otro objeto. Fue adoptada por Niklaus Wirth, autor del Pascal Original, e incorporada a otros de sus lenguajes como Component Pascal.

Componentes

Delphi proporcionó una buena implementación a la idea del uso de componentes, que son piezas reutilizables de código (clases) e interactúan con el EID (Entorno Integrado de Desarrollo) en tiempo de diseño y desempeñar una función específica en tiempo de ejecución. Desde un enfoque más específico de la herramienta, se catalogan como componentes todos aquellos objetos que heredan de la clase TComponent, donde se implementa la funcionalidad necesaria para interactuar con el entorno de desarrollo, la carga dinámica desde streams y la liberación de memoria mediante una jerarquía. Una gran parte de los componentes disponibles para Delphi son controles (derivados de TControl), que encapsulan los elementos de interacción con el usuario como botones, menús, barras de desplazamiento, etcétera.

Delphi incluye una biblioteca de clases bien diseñada denominada VCL (Visual Component Library, Biblioteca de Componentes Visuales) y, en sus versiones 6 y 7, una jerarquía multiplataforma paralela denominada CLX. Ésta también se incluye en Kylix. Las jerarquías de objetos incluyen componentes visuales y no visuales, tales como los pertenecientes a la categoría de acceso a datos, con los que puede establecerse conexiones de forma nativa o mediante capas intermedias (como ADO, BDE u ODBC) a la mayoría de las bases de datos relacionales existentes en el mercado. La VCL (Visual Component Library, Biblioteca de Componentes Visuales) también está disponible para el desarrollo en .NET.

Además de poder utilizar en un programa estos componentes estándar, es posible crear nuevos componentes o mejorar los ya existentes, extendiendo la funcionalidad de la herramienta. Existe un sinnúmero de componentes, tanto gratuitos como comerciales en la red, sin embargo la herramienta contiene un gran número de componentes.

Eventos

Delphi permite de manera sencilla ejecutar trozos de código en respuesta a acciones o eventos que ocurren durante el tiempo que un programa se ejecuta. Por ejemplo, cuando es presionado un botón, la VCL captura la notificación estándar de Windows, y detecta si hay algún método asociado al evento OnClick del botón. Si lo hay, manda ejecutar dicho método.

Los eventos pueden generarse debido a la recepción de señales desde elementos de hardware como el ratón o el teclado, o pueden producirse al realizar alguna operación sobre un elemento de la propia aplicación (como abrir un conjunto de datos, que genera los eventos BeforeOpen/AfterOpen). La VCL ha demostrado estar bien diseñada y el control que se tiene a través de los eventos de la misma es suficiente para la gran mayoría de aplicaciones.

Base de datos

Una de las principales características y ventajas de Delphi es su capacidad para desarrollar aplicaciones con conectividad a bases de datos de diferentes fabricantes. El programador de Delphi cuenta con una gran cantidad de componentes para realizar la conexión, manipulación, presentación y captura de los datos, algunos de ellos liberados bajo licencias de código abierto. Estos componentes de acceso a datos pueden enlazarse a una gran variedad de controles visuales, aprovechando las características del lenguaje orientado a objetos, gracias al polimorfismo.

En la paleta de componentes pueden encontrarse varias pestañas para realizar una conexión a bases de datos usando diferentes capas o motores de conexión.
Hay motores que permiten conectarse a bases de datos de diferentes fabricantes tales como BDE, DBExpress o ADO, que cuentan con manejadores para los formatos más extendidos.

Igualmente existen componentes de conexión directa para un gran número de bases de datos específicas como son: Firebird, Interbase, Oracle, etcétera, a continuación se muestra un breve resumen de las capas de conexión disponibles para las bases de datos más populares:

- Interbase/Firebird: IBX (InterBase eXpress), IBO (IB Objects), MDO (Mercury Data Objects), *DBExpress, BDE, FibPlus, Zeos
- Oracle: DOA (Direct Oracle Access), NCOci8
- dBase: BDE
- FoxPro: BDE
- Paradox: BDE
- Microsoft SQL Server: BDE, ADO, *DBExpress
- mySQL: Zeos (nativo), *DBExpress, BDE y ADO (usando ODBC)
- Postgres: BDE, ADO


Desarrollo visual

Como entorno visual, la programación en Delphi consiste en diseñar los formularios que componen al programa colocando todos sus controles (botones, etiquetas, campos de texto, etc.) en las posiciones deseadas, normalmente usando un ratón. Luego se asocia código a los eventos de dichos controles y también se pueden crear módulos de datos, que regularmente contienen los componentes de acceso a datos y las reglas de negocio de una aplicación.

Entorno Integrado de Desarrollo (EID)

También conocido como IDE por sus siglas en inglés (Integrated Development Environment), es el ambiente de desarrollo de programas de Delphi. Se trata de un editor de formularios (que permite el desarrollo visual), un potente editor de textos que resalta la sintaxis del código fuente, la paleta de componentes y el depurador integrado, además de una barra de botones y un menú que nos permite la configuración de la herramienta y la gestión de proyectos. En las ediciones Client/Server y Enterprise el EID también ofrece integración con una herramienta de control de versiones (PVCS).

Depurador integrado

Es una potente característica que permite establecer puntos de ruptura o también conocidos como breakpoints, la ejecución paso a paso de un programa, el seguimiento de los valores de las variables y de la pila de ejecución, así como la evaluación de expresiones con datos de la ejecución del programa. Con su uso, un programador experimentado puede detectar y resolver errores lógicos en el funcionamiento de un aplicativo desarrollado con Delphi. En las ediciones Client/Server y Enterprise se añade la opción de depuración a programas corriendo en equipos remotos (remote debugging), lo que posibilita el uso de todas las características del depurador con un programa ejecutándose en su entorno normal de trabajo y no en el equipo de cómputo del programador.

VENTAJAS Y DESVENTAJAS

Ventajas:

- Utiliza como lenguaje de programación Object Pascal, esto hace que sea intuitivo para muchos programadores con poca experiencia. Delphi es la primera herramienta en ofrecer un alto desempeño en código nativo compilado, con rapidez de ejecución y con la capacidad de acceso a bases de datos para cliente/servidor.
- Permite manejar casi cualquier tipo de base de datos que existe en el mercado, desde Oracle hasta MySQL pasando obviamente por Firebird, usa tanto componentes propios como ajenos de código abierto.
- Delphi ha evolucionando día con día y siempre ha sido compatible con todas las versiones de Windows (incluso con Kylix, con lo que todo el código desarrollado es compatible para algunos Linux y Unix). Actualmente la versión 8 soporta íntegramente la plataforma .Net.

Desventajas:

- Delphi cuenta con un punto negativo en relación a la desaparición de la compañía Borland, así como el tipo de ambiente que es (Low End Client), lo que lo hace poco competitivo en el desarrollo de aplicaciones de gran tamaño. No posee repositorio de componentes o facilidades de control de versiones.
- No utiliza sus propias rutinas de compilación se basa en el Framework de Microsoft.
- Desde un punto de vista negativo, Delphi se encuentra casi en extinción, debido a que Microsoft, con su producto Visual Studio, está avanzando y dejando atrás a Delphi dentro de la plataforma .NET pues en Win32 Delphi representa una plataforma sólida.
- Microsoft .Net trabaja sobre el RTM Framework 3.0, en el nuevo Visual Studio 2008 (con soporte completo para nuevas tecnologías de 3.0, como Workflow Foundation, Communications Foundation, Presentation Foundation y CardSpace,

además de agregar ADO.NET Entity Framework, LINQ, C# 3.0, VB.NET 9.0, LINQ to Objects API, mejoras en ClickOnce, soporte para desarrollo con Office System 2007, intellisense para javascript, XLinq y soporte para SQL Server Compact Edition, entre otras cosas.

- Con Delphi.net no es posible desarrollar aplicaciones para dispositivos móviles como ya que algunas partes necesarias de código necesario para ello aún no han sido liberadas por Microsoft.

2.3 CARACTERÍSTICAS, VENTAJAS Y DESVENTAJAS DE FIREBIRD

Firebird es un sistema de administración de base de datos relacional (o RDBMS) de código abierto, basado en la versión 6 de Interbase, cuyo código fue liberado por Borland en 2000. Su código fue reescrito de C a C++. El proyecto se desarrolla activamente y el 18 de abril de 2008 fue liberada la versión 2.1.

Tipos de servidor

Existen dos tipos de servidor Firebird para ser instalados: Classic y Super Server. Si bien tienen varias diferencias menores entre sí, la principal consiste en que el Super Server maneja hilos de ejecución individuales para cada conexión. Por lo tanto para un número reducido de conexiones el recomendado sería el Classic porque consumirá menor cantidad de recursos.

En caso de arquitecturas SMP, se debe utilizar el servidor Classic porque el Super Sever no tiene soporte para este tipo de arquitectura.

Los propios desarrolladores de Firebird recomiendan lo siguiente a la hora de decidirse por uno de estos servidores:

- En plataformas Windows seleccionar el Super server.
- En Linux simplemente elegir cualquiera, según las conexiones estimadas. En la mayoría de las situaciones no se notará diferencias en la ejecución.

Podría considerarse un tercer tipo, el Embedded (Incrustado). Éste consiste en una única biblioteca de enlace dinámico DLL (de unos 2 MB de tamaño) que contiene todo el servidor. De esta forma se puede tener un DBMS completo disponible y distribuible junto con aplicaciones de usuario sin requerir que este se instale por separado.

Características

- Es multiplataforma, y actualmente puede ejecutarse en los sistemas operativos: Linux, HP-UX, FreeBSD, Mac OS, Solaris y Microsoft Windows.
- Ejecutable pequeño, con requerimientos de hardware bajos.
- Arquitectura Cliente/Servidor sobre protocolo TCP/IP y otros.
- Soporte de transacciones ACID y claves foráneas.
- Es medianamente escalable.
- Buena seguridad basada en usuarios/roles.
- Diferentes arquitecturas, entre ellas el Firebird incrustado que permite ejecutar aplicaciones mono usuario en equipos de cómputo sin instalar el software Firebird.
- Bases de datos de sólo lectura, para aplicaciones que corran desde dispositivos sin capacidad de escritura, como cd-roms.
- Existencia de controladores ODBC, OLEDB, JDBC, PHP, Perl, .net, etc.
- Requisitos de administración bajos, siendo considerada como una base de datos libre de mantenimiento, al margen de la realización de copias de seguridad.
- Pleno soporte del estándar SQL-92, tanto de sintaxis como de tipos de datos.
- Completo lenguaje para la escritura de disparadores y procedimientos almacenados denominado PSQL.
- Capacidad de almacenar elementos BLOB (Binary Large OBjects).
- Soporte de User-Defined Functions (UDFs).
- Versión autoejecutable, sin instalación, excelente para la creación de catálogos en CD-Rom y para crear versiones de evaluación de algunas aplicaciones.

Mejoras del Lenguaje SQL

Se han realizado muchas mejoras al lenguaje SQL, incluyendo soporte a tablas derivadas (SELECT ... FROM (SELECT ... FROM)) con múltiples capas de anidamiento y la capacidad de unir conjuntos anidados, como se define en SQL200X.

También se ha agregado una nueva característica para ejecutar bloques de SQL procedural (PSQL) en instrucciones dinámicas de SQL, mediante la sintaxis de EXECUTE BLOCK (Bloque Ejecutable).

Una instrucción adicional (RETAIN) se agregó a la instrucción ROLLBACK de DSQL para hacerla consistente con COMMIT (RETAIN).

Todas las versiones de Firebird proveen dos modos transaccionales: NO WAIT (sin bloqueo, dando una excepción inmediatamente en un conflicto), y WAIT (con bloqueo, esperando hasta que termine la transacción en conflicto). Una nueva característica extiende el modo WAIT permitiendo un intervalo finito de espera y entonces reportando un error (isc_lock_timeout). Los intervalos de espera se especifican por transacción y están disponibles en la API usando la instrucción LOCK TIMEOUT y mediante la instrucción SET TRANSACTION.

Se trabajó para resolver los problemas de las vistas que son actualizables implícitamente, pero que tienen disparadores de actualización. Este es un cambio importante que puede afectar a los sistemas escritos para tomar ventaja de los comportamientos no documentados en versiones anteriores.

Los operadores de búsqueda de cadenas han sido re implementados y ahora trabajan correctamente con BLOBs de cualquier tamaño.

Desde Firebird 1.0 hasta la fecha, las operaciones de concatenación eran verificadas para un posible desborde en el momento de la preparación y dan una excepción si era posible un desborde de acuerdo a las longitudes declaradas de los operandos.

De esta manera, una expresión como "CAST('qwe' AS VARCHAR(30000)) || CAST('rty' AS VARCHAR(30000))" causaría una excepción debido a que 60,000 bytes excede el límite de las variables CHAR/VARCHAR. Ahora, esta condición solo causará una advertencia en el tiempo de preparación y será desplegada si la verificación en tiempo de ejecución detecta un desborde.

Algunas nuevas extensiones se han agregado al PSQL, incluyendo múltiples cursores explícitos, que están también disponibles dentro de las instrucciones de DSQL EXECUTE BLOCK.

El rastreo de invariantes en PSQL y la lógica de clonado de solicitudes se modificaron para corregir varios detalles de rendimiento y precisión con procedimientos recursivos.

Seguridad

El cifrado de las contraseñas usa un algoritmo (SHA-1), y el cifrado ahora está completamente basado en el servidor, y se requiere validarse con contraseña desde cualquier cliente remoto, sin importar los privilegios del usuario en la plataforma.

Los accesos de fuera del servidor a la base de datos de seguridad son rechazados, ya que el servidor rechazará cualquier intento de acceso a la base de datos de autenticación, incluso por la utilería GSEC del SYSDBA, con excepción de la API de Servicios.

El SYSDBA es el administrador de la base de datos de seguridad aunque los usuarios ahora pueden modificar sus propias contraseñas por medio de una vista que utiliza la nueva tabla RDB$USERS.

Los intentos para obtener el acceso al servidor usando técnicas de fuerza bruta en cuentas y contraseñas ahora son detectados y bloqueados, con la ayuda de la base de datos y la API de servicios.

Conexiones y redes

La capacidad de redirección del servidor (multi-hop), que fue deshabilitada por el desuso, ha sido restaurada y puesta a disposición mediante un parámetro de configuración para que sea utilizado con cuidado por quién entienda su utilización e implicaciones.

Firebird 2.0 reemplaza la implementación anterior del transporte local en Windows (conocida como IPC, o IPServer), con una nueva implementación, más robusta, llamada XNET. Funciona exactamente igual, brindando una manera eficiente para conectar el servidor localizado en la misma máquina con el cliente. Esta nueva implementación no sufre de la inestabilidad de su antecesor. Trabaja con el servidor Classic, para servicios no interactivos y sesiones de terminal, y elimina bloqueos cuando se intentan varias conexiones simultáneas. También se espera una ligera mejora en velocidad.

Cuando las versiones anteriores utilizaban el protocolo WNET (también conocido como NETBEUI), se realizaban solicitudes remotas en el contexto del token de seguridad del cliente. Ya que el servidor atiende cada conexión de acuerdo a sus credenciales de seguridad del cliente, esto significa que, si la máquina cliente está ejecutando algún usuario del sistema desde un dominio de NT, ese usuario requería permisos apropiados para acceder el archivo de base de datos físico, librerías UDF, etc., en el sistema de archivos del servidor.

En Firebird 2.0, las conexiones WNET se comportan de la misma manera que las TCP, sin tomar en cuenta los derechos de los usuarios del sistema operativo.

El manejo de conexiones en la arquitectura Super Server bajo POSIX ha sido mejorado y ahora utilizará SIGTERM y SIGINT para apagar todas las conexiones de manera natural.

Funcionamiento interno

Firebird 2.0 cuenta por primera vez con direccionamiento de 40 bits (64 bits internos) para superar el límite te tamaño de tablas de 30GB aproximadamente, impuesto por el direccionamiento de registros de 32 bits.

Recolección de basura: Desde Firebird 1.0 y versiones anteriores, la maquinaria Super Server ha realizado recolección de basura en procesos de fondo, abandonando el mecanismo de "cooperación" que maneja a la recolección de basura en la arquitectura Classic. Se ha reconocido que la recolección de basura cooperativa, si estuviese disponible, sería de utilidad para la arquitectura Super Server para algunas condiciones y configuraciones donde la espera para la recolección de basura en procesos de fondo cause cuellos de botella en el rendimiento.

En Firebird 2.0 Super Server, por omisión se realizan ambos tipos de recolección de basura (en proceso de fondo, y cooperativa). Para administrarlos, se encuentra un nuevo parámetro de configuración (GCPolicy) para poder configurar la recolección de basura para que sea sólo cooperativa o sólo en proceso de fondo.

La contención de bloqueos ha sido reducida tanto en el administrador de bloqueos como en el administrador de hilos de Super Server. También los volcados de memoria del administrador de bloqueos ahora cuentan con mejor información.

Se agregaron varias mejoras de depuración, incluyendo el nombre de archivo y la línea en los mensajes de registro (BUGCHECK). Se cuenta con nuevas utilerías de depuración y la capacidad de copiar los mensajes de sistema (syslog) a la consola del usuario.

La búsqueda de la carpeta/directorio raíz ha cambiado por lo que los procesos del servidor en Windows ya no hacen uso del registro, aunque las utilerías de línea de comandos aún lo utilizan.

UDFs, Utilerías y Herramientas de Línea de Comandos

Las funciones externas ahora tienen la capacidad de señalar un NULL SQL mediante un apuntador a NULL y la biblioteca nativa de funciones ib_udf se mejoró para permitir que las funciones de cadena ASCII_CHAR, LOWER, LPAD, LTRIM, RPAD, RTRIM, SUBSTR y SUBSTRLEN retornen nulos y sean interpretadas correctamente.

Muchas de las herramientas de línea de comando han sido reestructuradas y esta versión introduce las completamente nuevas herramientas de respaldo incremental NBak y Nbackup.

Los modos de shutdown (Detener) para un solo usuario y shutdown completo ahora son posibles mediante los nuevos parámetros STATE para los comandos gfix -shut y gfix -online.

Se eliminaron varios problemas de versiones anteriores de Firebird cuando se usaba el reporte de errores de gsec a través de la API de Servicios.

Ventajas

- No tiene costo la licencia.
- Se tienen las fuentes disponibles.
- Existen bastantes componentes de conexión para Delphi.
- Muchos administradores GUI.
- Bastante rápido.
- Una gran comunidad de Respaldo, incluso en español.
- Multiplataforma, Windows, Linux y Unix.
- Muy Buen manejo de concurrencia.
- Está hecho en C/C++.

Desventajas

- No hay muchas formas de optimizarlo.
- Por ser libre no tiene mucho soporte.
- Pocas funciones UDF's.
- Pocos manuales.

2.4 CARACTERÍSTICAS, VENTAJAS Y DESVENTAJAS DE ARQUITECTURA DE REDES

Una red de computadoras o también llamada red informática consta de dos o más computadoras conectadas entre sí que permiten compartir recursos, servicios e información, la cual suele consistir en archivos o datos. Los recursos son los dispositivos de almacenamiento de datos de una computadora compartida por otra mediante la red. Gracias a las redes se puede:

- Compartir programas y archivos.
- Compartir recursos.
- Compartir bases de datos.
- Utilizar software de red.
- Creación de grupos de trabajo.
- Gestión centralizada.
- Acceso a más de un sistema operativo.

Componentes de una red:

- Servidor.
- Estaciones de trabajo.
- Tarjetas de interfaz de red.
- Sistema de cableado.
- Recursos y periféricos compartidos.

Tipos de redes:

- Con servidor dedicado: como NetWare o Novell, donde uno o más equipos son servidores.
- Punto a punto: no existen máquinas intermedias, sólo sus extremos que normalmente son equipos de comunicaciones.

Para simplificar la comunicación entre programas (aplicaciones) de distintos equipos, se definió el Modelo OSI (figura 2.4.1) por la ISO (Organización Internacional para la Estandarización), el cual especifica 7 distintas capas de abstracción. Con ello, cada capa desarrolla una función específica con un alcance definido.

NÚMERO	CAPA	FUNCIÓN
7	Aplicación	Servicios de red a aplicaciones
6	Presentación	Representación de los datos
5	Sesión	Comunicación entre dispositivos de la red
4	Transporte	Conexión extremo-a-extremo y fiabilidad de los datos
3	Red	Direccionamiento lógico y determinación de la ruta
2	Enlace de Datos	Direccionamiento físico
1	Físico	Señal y transmisión binaria

Figura 2.4.1 Pila OSI

Dispositivos de Interconexión de Redes (figura 2.4.1):

- **Repeater (Repetidor)**. Dispositivo no inteligente que permite la interconexión de equipos y regenera señales en la red para llegar más lejos. Se utiliza en cableado lineal como Ethernet, actúa sobre el nivel más bajo de la jerarquía de protocolos y normalmente funciona a la misma velocidad de transmisión que las redes que conecta. Pueden propiciar un gran nivel de colisiones y tráfico de red. **Hub (concentrador)** es un repetidor multipuerto.
- **Bridge (Puentes)**. Dispositivo que conecta dos redes a nivel capa de enlace, con el mismo protocolo y medio de transmisión. Puede determinar la dirección fuente y el destino sobre la red para permitir o denegar el paso de paquetes de datos.

- **Router (Ruteadores o Enrutador)**. Dispositivo utilizado para interconectar redes que usan el mismo protocolo, sobre el mismo o diferente medio físico de comunicación. Pueden usarse para ligar geográficamente redes dispersas al mismo tiempo. Examina **frames (marco o paquete, bloque fijo de datos transmitidos como una sola entidad)**[3] sobre la red y si están destinados a otra, consulta su tabla de ruta la cual contiene una lista de nodos y rutas entre éstos, y usando un algoritmo específico de ruteo obtiene la mejor ruta. Similares a los bridges pueden operar en modo local o remoto, o soportar ambos, pero hacen uso de la dirección física[4].
- **Gateway (Pasarelas)**. Dispositivo que interconecta sistemas de redes diferentes en tiempo real entre un usuario y una de las varias estaciones remotas. Muestra transparencia al usuario de las conexiones que realiza.
- **Host (Estaciones de trabajo)**. Dispositivo con un nombre único que ofrece servicios a otros ordenadores conectados a la red. Puede ser una computadora, un servidor, un dispositivo de almacenamiento por red, etc. Este **hostname (nombre de equipo)**, ayuda al administrador de la red a identificar las máquinas sin tener que memorizar una dirección IP para cada una de ellas.
- **Switch (Conmutador)**. Dispositivo que trabaja a nivel capa enlace de datos, como los bridges, permiten interconexiones de múltiples segmentos físicos de la red local en una sola de gran tamaño. Envían y contribuyen el tráfico con base en sus direcciones de control de acceso al medio.

[3] http://es.wikipedia.org/wiki/Frame

[4] Stalling, William. “Comunicaciones y redes de computadoras”. Prentice Hall, 6º Ed.

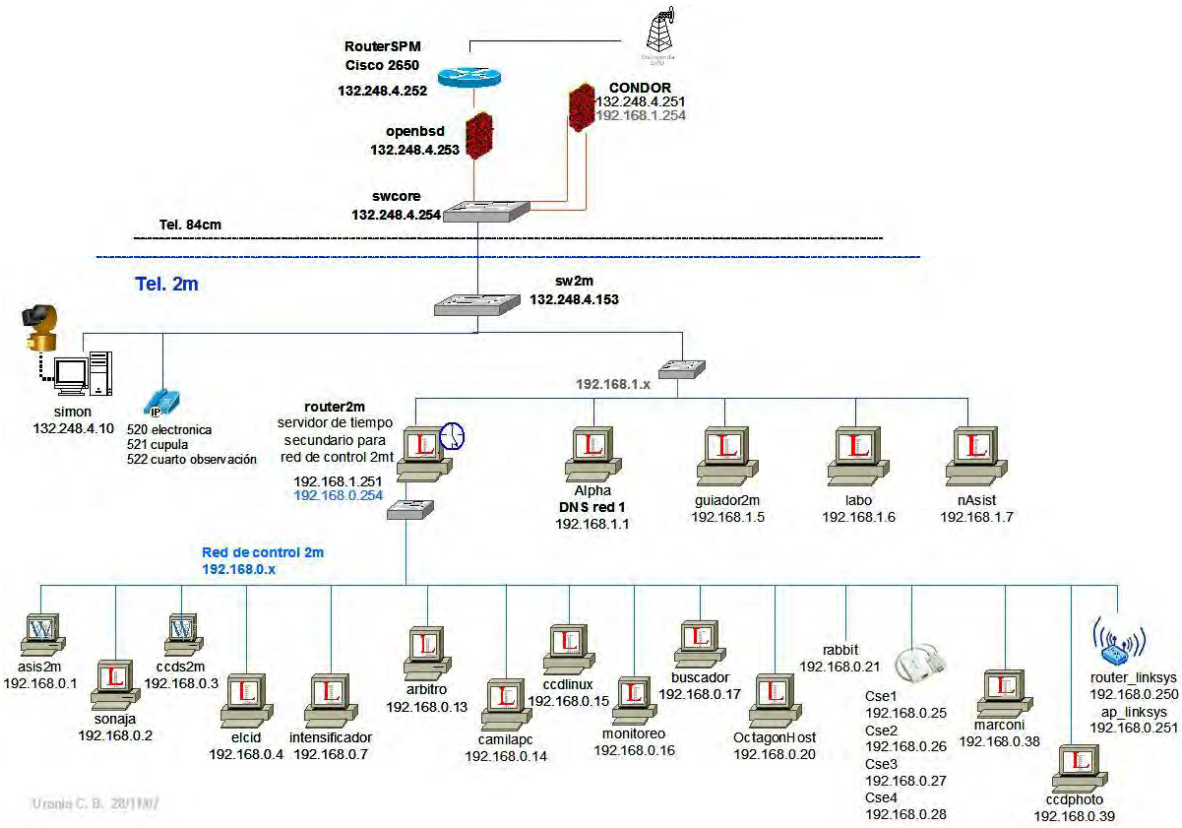


Figura 2.4.2 Diagrama de una red[5]

Tipos de redes según su extensión geográfica

Local Area Network (LAN, Redes de Área Local): Es un sistema de interconexión de equipos, basado en líneas de velocidad en un área privada restringida[6]. Las principales tecnologías usadas son: Ethernet, Token Ring (Red en Anillo), ARCNET y FDDI (Fiber Distributed Data Interface, Interface de Datos Distribuidos por Fibra).

[5] http://www.astrossp.unam.mx/computo/red2m.jpg

[6] http://sistemas.itlp.edu.mx/tutoriales

Ventajas y desventajas

- Restricción mínima y máxima en la extensión del alcance.
- Restricción máxima de velocidad (decenas o cientos de megabits por segundo).
- Restricción en número de nodos (de treinta a cincuenta).
- Capacidad de conexión con otras redes locales o de área extensa.
- Propiedad privada.
- Presenta baja tasa de error en transmisiones de datos.
- Permite la integración en la misma red de una gran cantidad de dispositivos.
- La fibra óptica ofrece un medio seguro, confiable y libre de ruido.

Metropolitan Area Network (MAN, Redes de Área Metropolitana): Es un sistema de interconexión de equipos distribuidos en una zona que abarca diversos edificios, por medios pertenecientes a la misma organización propietaria de los equipos. Se utiliza normalmente para interconectar redes de área local.

Ventajas y desventajas:

- Integración de varios puntos en un mismo enlace.
- Posibilidad de incremento hacia otros puntos para integración de la misma red.
- Transmisión multimedia.
- Servicio de banda ancha.
- Permite superar los quinientos nodos de red.
- Permite la distancia entre nodos de varios kilómetros.
- Restricción máxima en la extensión del alcance.
- Tiempos de acceso mínimos y servicios síncronos para aplicaciones en tiempo real.
- La fibra óptica ofrece un medio seguro, confiable y libre de ruido[7].

[7] http://canalhanoi.iespana.es/informatica/redman

Wide Area Network (WAN, Redes de Área Extensa): Es un sistema de interconexión de equipos geográficamente dispersos, incluso a nivel continental. El sistema de conexión normalmente involucra a redes públicas.

Ventajas y desventajas:

- Pueden utilizar un software especializado para incluir mini o macrocomputadoras como elementos de red.
- No se limita al espacio geográfico para establecer comunicación.
- Puede utilizar enlaces a satélites.
- La fibra óptica ofrece un medio seguro, confiable y libre de ruido.
- Los equipos deben poseer gran capacidad de memoria para un acceso rápido.
- Poca seguridad en los equipos (virus y troyanos entre otros).

Su clasificación puede resumirse en la siguiente tabla.

TIPO	DISTANCIA	ALCANCE
LAN	10 m	Habitación
	100 m	Edificio
	1 km	Localidad
MAN	50 km	Ciudad
WAN	100 km	País
	1000 km	Continente
INTERNET	10,000 km	Planeta

Figura 2.4.3 Clasificación de redes geográficamente

Modos de Transmisión

Un método de caracterizar líneas, dispositivos terminales, computadoras y módems es por su modo de transmisión o de comunicación. Las tres clases de modos de transmisión son:

Transmisión Simplex (sx). También llamada unidireccional es aquella que ocurre en una dirección solamente, deshabilitando al receptor de responder al transmisor. Normalmente la transmisión simplex no se utiliza donde se requiere interacción humano-máquina. Figura 2.4.4

Figura 2.4.4 Transmisión Simplex

Transmisión Half-Duplex (hdx). Permite transmitir en ambas direcciones; sin embargo, la transmisión puede ocurrir solamente en una dirección a la vez. Tanto transmisor y receptor comparten una sola frecuencia. Figura 2.4.5

Figura 2.4.5 Transmisión Half-Duplex

Transmisión Full-Duplex (fdx). Permite transmitir en ambas direcciones, simultáneamente por el mismo canal. Existen dos frecuencias una para transmitir y otra para recibir. Figura 2.4.6

Figura 2.4.6 Transmisión Full-Duplex

Arquitectura de Red

Se define por su topología, el método de acceso a la red y protocolos de comunicación.

Topología de red o forma lógica de red es el arreglo físico de nodos y los medios de transmisión y conexión dentro de la estructura de red. La cuestión más importante al considerar la elección del sistema de cableado es su costo, igualmente su rendimiento total y su integridad. Existen distintos tipos de topologías:

Configuración Bus. Tiene un único canal de comunicaciones denominado bus troncal el cual se conecta a los diferentes dispositivos, compartiendo el mismo canal para comunicarse entre sí (ver figura 2.4.7). Físicamente cada host está conectado a un cable común, por lo que se pueden comunicar directamente. La ruptura del cable hace que los hosts queden desconectados al no haber ninguna otra conexión entre sí. Son de fácil implementación y crecimiento, económicas, son de longitud limitada, el desempeño disminuye a medida que crece la red, el canal requiere ser correctamente cerrado y altas pérdidas en la transmisión debido a colisiones entre mensajes.[8]

Figura 2.4.7 Topología de bus

Configuración Anillo. Los datos se transmiten a lo largo del anillo y cada computadora conectada a la siguiente y la última a la primera, examina los datos para determinar si van dirigidos a ella. De no ser así, los transmiten a la siguiente computadora (ver figura

[8] http://es.wikipedia.org/wiki/Arquitectura_de_red

2.4.8). Cada estación tiene un receptor y un transmisor que hace la función de repetidor. Este proceso se repite hasta que los datos llegan a su destino. Una red en anillo permite la transmisión simultánea de múltiples mensajes, pero como varias computadoras comprueban cada mensaje la transmisión de datos resulta más lenta[9]. La comunicación se da por el paso de un token o testigo evitando eventuales pérdidas de información debidas a colisiones. Si algún nodo de la red deja de funcionar, la comunicación en todo el anillo se pierde. En un anillo doble, dos anillos permiten que los datos se envíen en ambas direcciones creando esta configuración redundancia (tolerancia a fallos). Es una topología simple, de fácil implementación y crecimiento. Las longitudes del canal son limitadas y el canal usualmente se degrada a medida que la red crece.

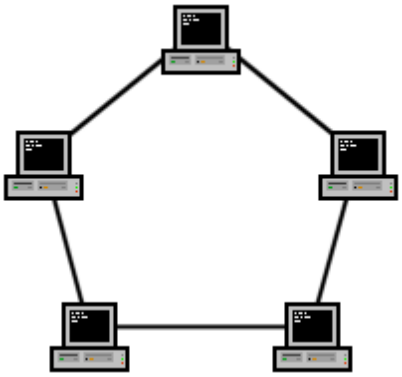


Figura 2.4.8 Topología de anillo

Configuración Estrella. Se conectan todos los nodos a un concentrador central el cual envía todas las transmisiones recibidas de cualquier nodo periférico a todos los nodos de la red, incluso a veces de regreso al nodo que envió. Todos los nodos periféricos se pueden conectar con los demás transmitiendo o recibiendo del nodo central únicamente (ver figura 2.4.9). Un fallo en la línea de conexión de cualquier nodo con el concentrador

[9] http://www.arqhys.com/construccion/redes-arquitectura

provocaría el aislamiento de dicho nodo respecto a los demás, pero el resto permanecería intacto. La mayoría de las redes de área local que tienen un router, un switch o un hub siguen esta topología, los cuales serían el nodo central por el que pasan todos los paquetes. La desventaja radica en la carga que recae sobre éstos, la cantidad de tráfico que deberá soportar es grande y aumentará conforme se agreguen más nodos periféricos, lo que la hace poco recomendable para redes de gran tamaño. Además, un fallo en el nodo central puede dejar inoperante a toda la red. Esto último conlleva también una mayor vulnerabilidad de la red, en su conjunto, ante ataques. Es costosa.

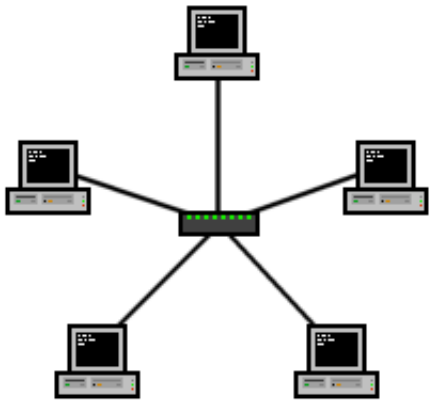


Figura 2.4.9 Topología de estrella

Configuración de Árbol. También conocida como topología jerárquica, llamada así por su apariencia estética puede verse como una colección de redes en estrella ordenadas en jerarquía. Se tienen nodos periféricos individuales que requieren transmitir y recibir, y no necesitan actuar como repetidores o regeneradores. La función del nodo central se puede distribuir. Los nodos individuales pueden quedar aislados de la red por un fallo puntual en la ruta de conexión. Si falla un enlace que conecta con un nodo individual, este queda aislado; si falla un enlace con un nodo no individual, la sección entera queda aislada del resto. Para la cantidad de tráfico de red que se necesita para retransmitir a todos los nodos se desarrollaron nodos centrales más avanzados que

permiten mantener un listado de las identidades de los diferentes sistemas conectados a la red llamados switches de red, que “aprenderían” cómo es la estructura de la red transmitiendo paquetes de datos a todos los nodos y luego observando de dónde vienen los paquetes respuesta (ver figura 2.4.10). Puede comenzar con la inserción del servicio de internet del proveedor, pasando por el router, luego por el switch y derivar a otro switch o router, o sencillamente a los hosts. Desde el primer router que se tiene se ramifica la distribución de internet dando lugar a la creación de nuevas redes y/o subredes tanto internas como externas.[10]

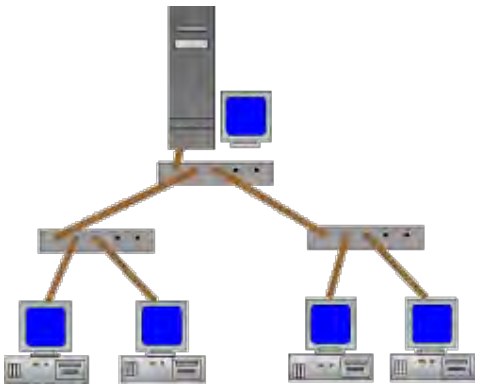


Figura 2.4.10 Topología de árbol

Configuración en Malla. Hay al menos tres nodos donde cada uno está conectado a todos los demás. El número de caminos arbitrarios en las redes de malla las hace más difíciles de diseñar e implementar, pero su naturaleza descentralizada las hace muy útiles pues no puede existir absolutamente ninguna interrupción en las comunicaciones. Una red totalmente conectada o completa es una topología de red en la que hay un enlace directo entre cada pareja de nodos disponible (ver figura 2.4.11). En una red totalmente conectada hay $\frac{n \times (n-1)}{2}$ enlaces directos. Este tipo de redes son de alto costo pero ofrece una redundancia y fiabilidad superiores gracias a las múltiples rutas a

[10] http://es.wikipedia.org/wiki/Arquitectura_de_red

seguir. Es una opción aplicable a las redes Wireless (inalámbricas) y a las redes Wired (cableadas).

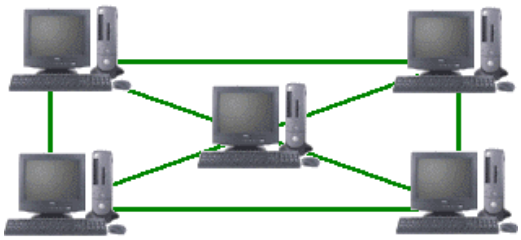


Figura 2.4.11 Topología de Malla

2.5 CARACTERÍSTICAS, VENTAJAS Y DESVENTAJAS DE APLICACIONES CLIENTE/SERVIDOR

Las aplicaciones cliente/servidor basan su lógica de comunicación en la arquitectura o modelo cliente/servidor, éste modelo especializa cada programa o equipo de cómputo, también llamados componentes, con el desarrollo de una tarea específica, es un procesamiento cooperativo donde cada uno de los componentes pide servicios a otro, lo anterior tiene como propósito que cada tarea en la aplicación la realice con la mayor eficiencia posible. Éste modelo proporciona al usuario final un acceso transparente a las aplicaciones, datos, servicios de cómputo o cualquier otro recurso.

Las aplicaciones cliente/ servidor se generalizan en dos entidades, un servidor que es el que ofrece un servicio y un cliente que es quién pide ese servicio. Los usuarios invocan la parte del cliente de la aplicación, ésta construye una solicitud para ese servicio y realiza un envío por el canal de comunicación hacía el servidor, el servidor recibe la solicitud de servicio, procesa la información y devuelve los resultados en forma de respuesta, generalmente un servidor puede tratar múltiples peticiones (múltiples clientes) al mismo tiempo (ver figura 2.5.1).

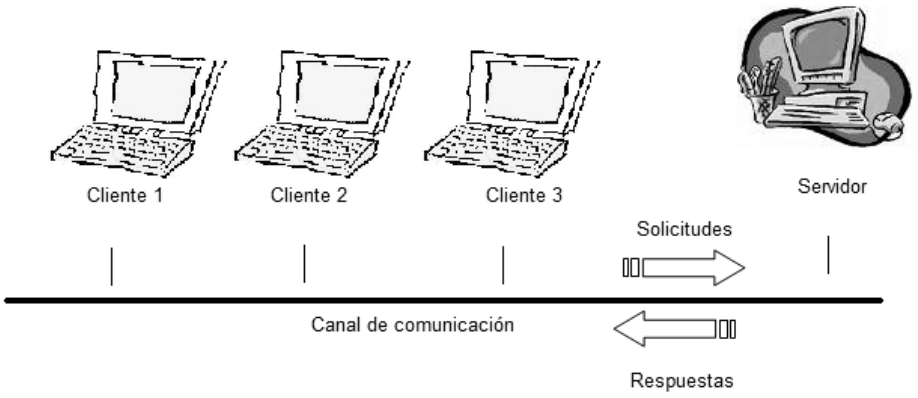


Figura 2.5.1 Aplicaciones Cliente/Servidor

Para que exista comunicación entre los clientes y los servidores debe existir un canal de comunicación que proporcione los mecanismos básicos de transporte y direccionamiento.

Características del modelo Cliente/Servidor

- El cliente y el servidor pueden actuar tanto como una sola entidad como entidades separadas.
- Las funciones de Cliente y Servidor pueden estar en plataformas separadas o en la misma plataforma, por ejemplo, cliente (Windows), servidor (Linux), como las entidades comparten el mismo protocolo de comunicación, no existen problemas entre las múltiples plataformas que se utilicen.
- Un servidor puede dar servicio a múltiples clientes en forma concurrente.
- Un sistema de servidores realiza múltiples funciones al mismo tiempo, esto representa una imagen de un solo sistema en los clientes.

Tipos de clientes en el modelo

1. “Cliente flaco”
    - Servidor generalmente saturado.
    - Gran cantidad de datos en el canal de comunicación.
2. “Cliente gordo”
    - La mayoría del trabajo se encuentra del lado del cliente.
    - No hay una centralización en la gestión de la BD.
    - Gran cantidad de datos inútiles en el canal de comunicación.

Ver figura 2.5.2

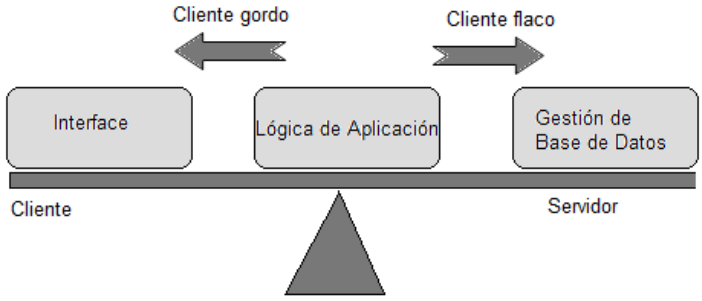


Figura 2.5.2 Tipos de clientes en el modelo Cliente/Servidor

Tipos de Servidores en el modelo

- **Servidores de archivos.** Servidor donde almacena archivos y aplicaciones, como por ejemplo procesadores de texto, hojas de cálculo, etc.
- **Servidores de bases de datos.** Servidor donde se almacenan las bases de datos, tablas, índices.
- **Servidores de objetos.** Servidor que contiene objetos que deben estar fuera del servidor de base de datos, estos objetos pueden ser videos, imágenes u objetos multimedia.
- **Servidores Web.** Servidores que son utilizados para la comunicación entre diferentes entidades web.
- **Servidores de software de grupo.** Los servidores de software de grupo permite organizar el trabajo de un grupo. El servidor gestiona los datos que dan soporte a dichas tareas.
- **Servidores de correo.** Gestiona el envío y recepción de correo de un grupo de usuarios.
- **Servidor de impresión.** El servidor de impresión administra las solicitudes de impresión de los clientes.

Aplicación Cliente/Servidor una capa

Las aplicaciones cliente/servidor de una capa agrupan toda la lógica en una sola entidad, (lógica de presentación, lógica de aplicación y fuente de datos), la aplicación se comporta como cliente y servidor simultáneamente. (Figura 2.5.2)

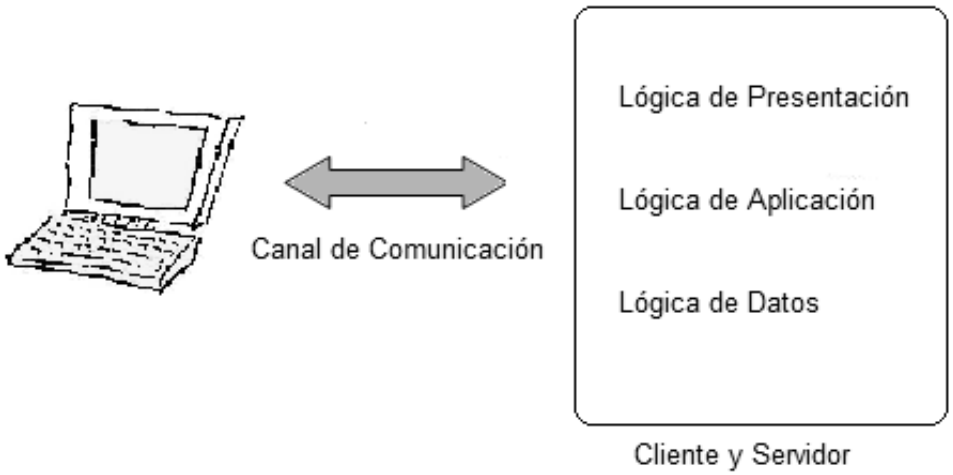


Figura 2.5.2 Aplicación Cliente/Servidor una capa

Aplicación Cliente/Servidor dos capas

Generalmente las aplicaciones Cliente/Servidor están basadas en modelos de dos capas. El modelo de dos capas consta de un cliente, que mantiene la lógica de la aplicación y un servidor que administra las peticiones a la base de datos. Éste tipo de aplicaciones son generalmente utilizadas cuando se requiere poco procesamiento de datos, además de que la organización tiene una base de datos centralizada en un solo servidor. Figura 2.5.3.

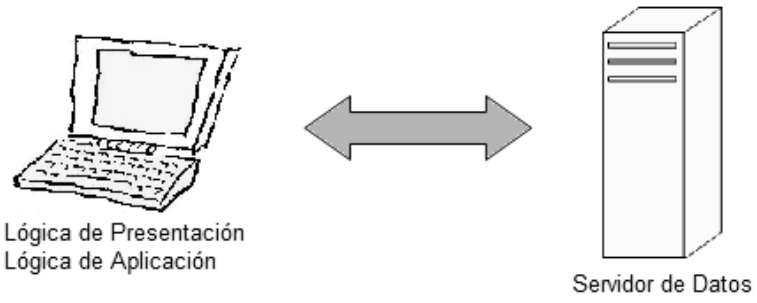


Figura 2.5.3 Aplicación de Cliente/Servidor dos capas

Aplicación Cliente/Servidor tres capas

Las aplicaciones en tres capas es un modelo que evolucionó del modelo de dos capas, para éste modelo se agregó una capa intermedia entre el cliente y el servidor de base de datos. La capa que se agregó consiste en un servidor de aplicaciones que contiene la lógica de la aplicación. Generalmente éste tipo de aplicaciones se utiliza cuando se requiere mucho procesamiento de datos en la aplicación, además de que los procesos no se encuentren relativamente relacionados con los datos, a su vez se requiera aislar la tecnología de base de datos para que sea fácil realizar cambios. Figura 2.5.4.

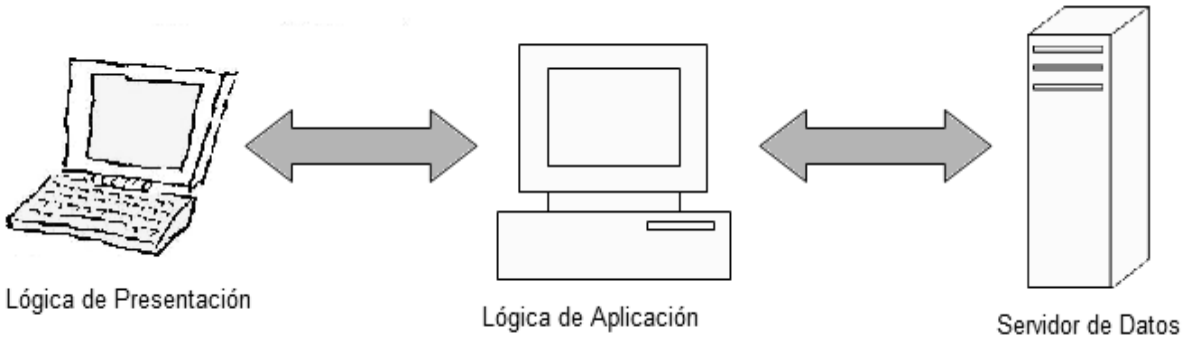


Figura 2.5.4 Aplicación Cliente/Servidor tres capas

Aplicación Cliente/Servidor 'n' capas

En las aplicaciones Cliente/Servidor 'n' capas, la lógica de presentación, la lógica de la aplicación y el servidor de base de datos, además de estar separados, pueden ser subdivididos aún más. Éste modelo ha predominado en el desarrollo de aplicaciones multiplataforma. Figura 2.5.5.

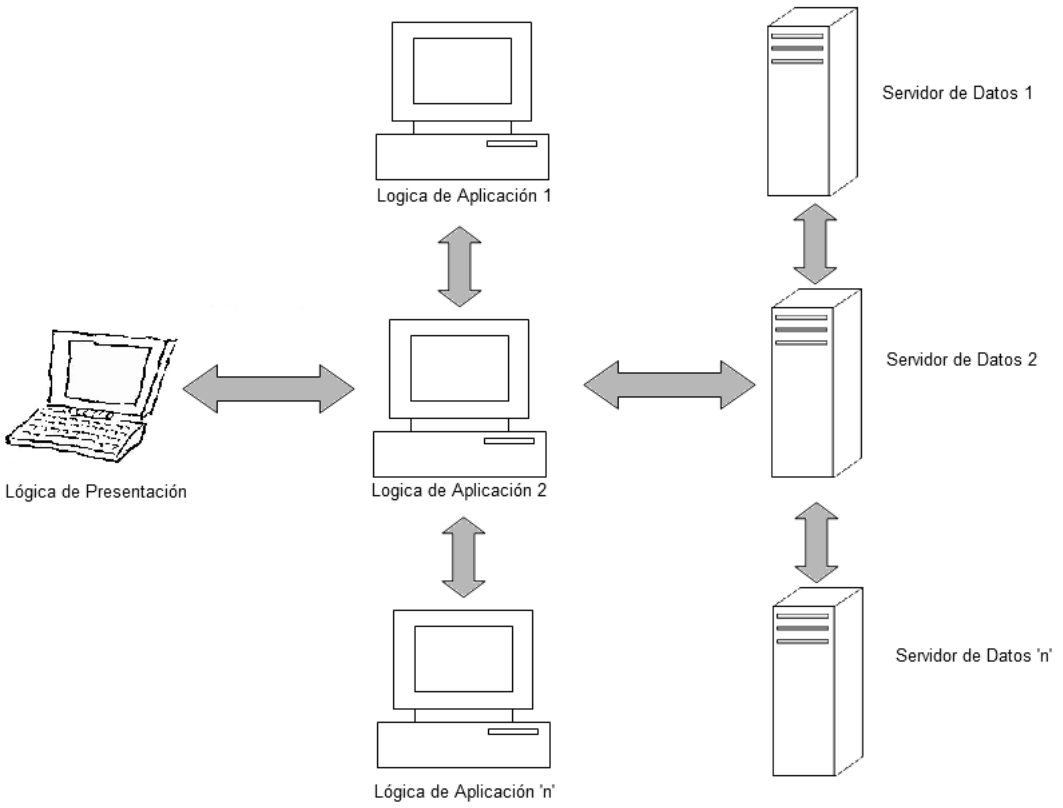


Figura 2.5.5 Aplicación Cliente/Servidor 'n' capas

Ventajas de aplicaciones Cliente/Servidor

El cambio radical en los modelos de computación, desde los sistemas monolíticos hasta los modelos cliente/servidor, que las empresas de tecnología en los últimos años han adoptado es la especialización de tareas. Las aplicaciones cliente/servidor especializa cada tarea, y una tarea puede ser subdivida en más tareas o procesos. Las ventajas de éste tipo de aplicaciones se listan a continuación.

- El desarrollo del software puede ser realizado de manera paralela, lo que implica menos tiempo de desarrollo.
- Mejora el rendimiento. Los datos son almacenados y usados donde son generados, lo cual permite distribuir la complejidad del sistema en los diferentes sitios de la red, optimizando la labor de cada uno de ellos.
- Mantenimiento y soporte más sencillo.

- Mayor flexibilidad. Pueden ser agregados nuevos módulos para dotar al sistema de nuevas funcionalidades sin tener que modificar módulos existentes.
- Alta escalabilidad. La principal ventaja es la escalabilidad, se refiere al manejo de múltiples peticiones con el mismo rendimiento.
- El tráfico en los canales de comunicación se reduce. Los clientes únicamente se comunican con otros módulos lo estrictamente necesario, obtiene los datos solicitados y cierra la conexión, por lo que deja la red libre para realizar más conexiones.

Éste modelo ha sobrevivido a los grandes cambios en los paradigmas de programación a lo largo de los años. El no estar atado algún lenguaje de programación garantiza una fácil adaptación por parte del equipo de desarrollo. Permitiendo su implementación en .net, java, php, etc. Además de que su principal fortaleza es que puede convivir con aplicaciones creadas en diferentes lenguajes.

Desventajas de aplicaciones Cliente/Servidor

- Costo elevado. El costo de la administración es elevado debido a la complejidad técnica del servidor.
- El servidor es el eslabón débil. El servidor es el único eslabón débil en las aplicaciones cliente/servidor, debido a que toda la red está construida en torno a él.
- Manejo óptimo de errores. Se deben de tener estrategias en el manejo de errores para mantener la consistencia de los datos y de las aplicaciones.
- Las distancias en el canal de comunicación puede contribuir a la velocidad de respuesta de la información.
- Consistencia en los datos. El compartir los mismos datos para diferentes aplicaciones puede hacer que los datos no sean consistentes.

CAPÍTULO 3

3.1 PROBLEMÁTICA ACTUAL

En este capítulo describiremos el proceso general común que se lleva a cabo para la consulta, préstamo y entrega de material gráfico en una biblioteca, así como las dificultades que se presentan durante el mismo.

En el diagrama general del proceso Solicitud, que se muestra en la figura 3.1.1, el usuario realiza una petición de material y espera una resolución satisfactoria que cubra sus necesidades.

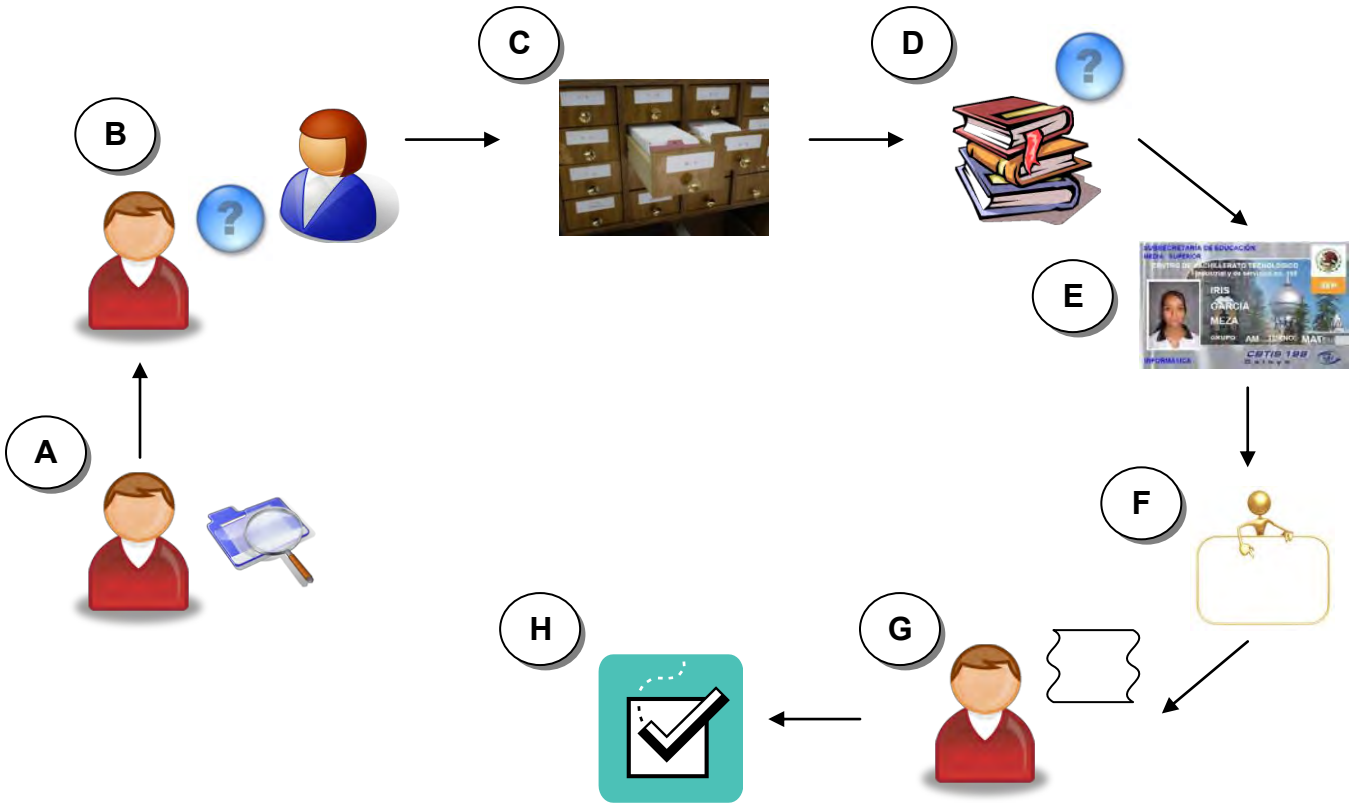


Figura 3.1.1 Diagrama de proceso Solicitud

A. El proceso inicia cuando el usuario acude a la biblioteca para la consulta o solicitud de préstamo domiciliario de algún material.

Durante este proceso suelen presentarse inconvenientes tales como:

B. El usuario sólo puede realizar la petición de prestación local y personalmente en la biblioteca, enfrentando largas filas de espera para realizar algún trámite Si el usuario solicita ayuda puede toparse con problemas como: la falta de auxilio por parte del personal, ocasionada por la mala organización del grupo y área de trabajo, falta de capacitación, registro inexacto de préstamos, o la localización inexacta del material en los estantes.
C. Al realizar una búsqueda, el usuario o el bibliotecario mismo, pueden tropezarse con el uso de ficheros complejos y anticuados que resten tiempo y confundan su exploración. Búsquedas de carácter exclusivo impidiendo el acceso a registros de otras entidades de la institución, o generales que no especifiquen la ubicación física exacta en más de una localidad.
D. Una vez localizado el material existe la posibilidad que el número de ejemplares sea insuficiente debido a la falta de actualización en los catálogos, no correspondiendo con las altas del nuevo y bajas del inexistente, ocasionando poca o nula disponibilidad para su préstamo domiciliario; que el material se encuentre prestado y el usuario desconoce cuándo será devuelto para realizar la petición inmediata a su entrega.
E. Al realizar la solicitud de préstamo es necesario el uso de una credencial emitida por la institución, siendo ésta la única manera de realizar el trámite. Otro conflicto es la actualización necesaria y vigente de dicha credencial, trámite realizado en dependencias ajenas a la biblioteca como servicios escolares.
F. Al momento de la solicitud el usuario puede toparse con normas bibliotecarias que impiden el préstamo de manera independiente por cada biblioteca de la institución donde está dado de alta, acotando el número de posibles préstamos

que pudiera requerir. En caso de ser posible y solicitar préstamos externos, el usuario debe realizar el trámite por su cuenta debido a que no existen préstamos inter – bibliotecarios.

G. Para las solicitudes de préstamo de material restringido el usuario debe llenar por escrito formatos muy largos con demasiados campos y que consumen mucho tiempo, una por cada ejemplar que se solicite.

H. El proceso finaliza cuando el usuario satisface peticiones de préstamo y se le otorga el material.

En el diagrama general de proceso Entrega, que se muestra en la figura 3.1.2, el usuario realiza la devolución de material.

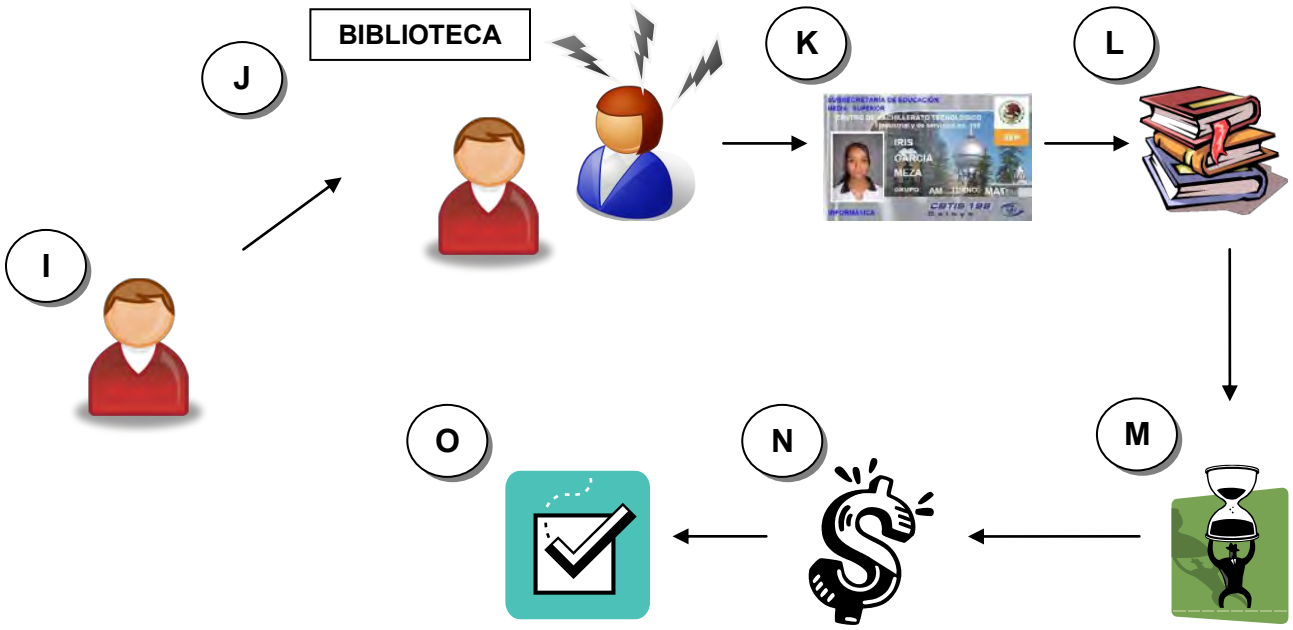


Figura 3.1.1 Diagrama de proceso Entrega

I. El proceso inicia cuando el usuario acude a la biblioteca para la devolución de material.

Durante este proceso suelen presentarse dificultades tales como:

J. El usuario sólo puede realizar la renovación del tiempo de préstamo de manera local y personalmente en la biblioteca. Al acercarse al módulo de préstamos puede enfrentarse con una atención deficiente por parte del encargado: el bibliotecario no se encuentra en su área de trabajo, los registros de préstamos y devoluciones son poco confiables, recibir un trato poco amable o un servicio prolongado que le reste demasiado tiempo.
K. Al realizar la entrega es necesario el uso de la credencial emitida por la institución para garantizar la correlación usuario-material, haciendo de esta la única manera de realizar el trámite.
L. Dado el caso que los kárdex de los libros que registran la fecha de entrega proporcionada por el bibliotecario carezcan de renovación constante, pueden provocar la confusión del usuario en el tiempo de entrega del material.
M. Las fechas de devolución asentadas en el kárdex son períodos específicos de tiempo de entrega, los cuales son a veces largos en consideración con las necesidades del usuario, sin permitirle la devolución antes.
N. En el caso de tener multas el usuario debe realizar el pago en dependencias ajenas a la biblioteca, haciendo poco ágil esta comisión.
O. El proceso finaliza cuando el usuario ha devuelto el material facilitado y puede volver a realizar cualquier solicitud de préstamo.

Haciendo alusión a las etapas que enuncian los escritores americanos R. David Lankes y Abby S. Kasowitz[11] en algunos de sus trabajos, las cuales sugieren un proceso de seis pasos para ayudar a que cualquier organización pueda crear, desarrollar y operar sus servicios digitales, lo anterior con el fin de hacer eficiente cualquier proceso, se

[11] Lankes, R. David, Kasowitz, A. S. **"The AskA starter kit: how to build and maintain digital references services"**, Syracuse University. NY, 1998

tomarán como parámetro para señalar las deficiencias existentes en el servicio actual que se proporciona en las bibliotecas, las cuales se cubrirán con el sistema propuesto para la optimización del proceso. Dichas etapas son:

1. Información. Investigación preliminar sobre los servicios existentes en el campo de especialización del servicio a diseñar.

2. Planeamiento. Políticas, procedimientos y métodos que se deben desarrollar para asegurar la organización del servicio.

3. Entrenamiento. Desarrollo de un plan de entrenamiento y preparación del personal a realizar la labor; incluye los materiales, actividades y herramientas necesarias.

4. Prototipos. Desarrollar el servicio primeramente a modo de prueba, para identificar fallos y corregirlos a tiempo.

5. Contribución. Importancia del desarrollo del servicio para la organización, desarrollando a tales efectos una publicidad continuada para apoyar el servicio.

6. Evaluación. Proveer evaluaciones regulares para asegurar la calidad del producto o servicio y conocer las ventajas reportadas a la organización por la implantación del mismo

3.2 REQUERIMIENTOS GENERALES Y PARTICULARES

Requerimientos generales

Se pretende que la aplicación funcione en un equipo de cómputo central al que se puedan conectar los lectores desde otros equipos de cómputo para realizar consultas.

Las consultas permitirán acceder a la información sobre los libros que estén disponibles y que posean una determinada palabra en su título, o que pertenezcan a un cierto autor. Por otra parte el encargado de la biblioteca se puede conectar también desde el terminal que posee en su lugar de trabajo para gestionar el préstamo y devolución de libros. Es de especial interés que la aplicación controle las fechas de préstamo y devolución. Además el encargado podrá dar de alta o de baja los libros que existan en la biblioteca. Adicionalmente se ha decidido desarrollar una interfaz para la aplicación que sea fácil de reutilizar y modificar. La interfaz debe estar formada por los siguientes elementos: menús, ventanas de diálogos y ventanas de salida. El menú permitirá al usuario elegir entre varios submenús. Los submenús a su vez estarán formados por opciones o ítems. Las diversas opciones permitirán ejecutar las funciones o acciones de la aplicación que utilice la interfaz. Las ventanas de diálogos servirán para que el usuario introduzca una información de entrada en la aplicación. Un diálogo tendrá una o más líneas de entrada para introducir información.

Las ventanas de salida permitirán mostrar resultados y mensajes a los usuarios. Se pueden tener abiertas varias ventanas, cada una de ellas identificada por un título.

La figura 3.2.1 muestra el diagrama de contexto del sistema de Gestión de Biblioteca, destacando los actores en torno al sistema. Se puede ver que existen Lectores y Encargado. Estos actores representan los roles que juegan las personas que interactúan con el sistema.

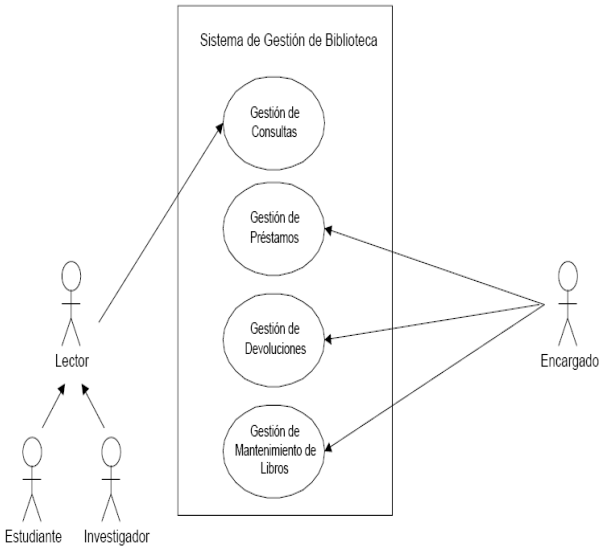


Figura 3.2.1 Diagrama de Contexto del Sistema de Biblioteca

Requerimientos funcionales particulares

- Debe ser un sistema completamente integrado, capaz de articular y gestionar diferentes módulos funcionales, diferentes repositorios de datos, grupos de usuarios y servicios de manera consistente.
- Debe soportar un incremento del 80% en el tamaño de las bases de datos en relación al tamaño actual, en la infraestructura de servicios y en el caudal de transacciones sin requerir de una actualización en la configuración inicial de hardware.
- Debe permitir la confección e implementación de parámetros de gestión de servicios puntuales según colecciones, unidades administrativas, tipos de material y grupos de usuarios.
- Debe permitir la confección e implementación de parámetros de registros y gestión catalográfica puntuales según colecciones, unidades administrativas, tipos de material, grupos de usuarios y usuarios específicos.
- Debe soportar las siguientes funciones generales: catálogo público de acceso en línea, servicios catalográficos completos (registros bibliográficos, ejemplares, control de autoridades, gestión de repositorios digitales), servicios documentales

completos (circulación, servicios de alerta, impresión y distribución de documentos), adquisiciones (pedidos, reclamos, recepción, pagos, gestión de proveedores y gestión financiera), control de existencias (check-in, reclamos, inventario), gestión de usuarios y gestión integrada (estadísticas, auditoría de existencias y procesos, procedimientos de respaldo, procedimientos de recuperación de datos).

- Debe contar con entornos de prueba y entrenamiento separados del entorno de producción.
- Debe ser capaz de importar y exportar datos (registros bibliográficos, de autoridades, etc.) a través de diversos esquemas de comunicación o esquematización de datos (ejemplo MARC 21, Dublin Core) y diversos formatos de codificación (XML, ISO 2709, texto plano, etc.).
- Debe soportar cambios en la definición de los formatos de registro.
- Debe permitir realizar procesos masivos por lotes (batch) y en tiempo real.
- Debe permitir realizar en tiempo real, de manera interactiva y continua, procesos de creación, actualización, acceso y mantenimiento de todos los datos gestionados por el sistema.
- Debe proveer de reportes actualizados en tiempo real y continuo.
- Debe disponer de procedimientos de respaldo continuo y recuperación de datos para todas las transacciones contempladas en el sistema.
- Debe incluir herramientas para el diseño y generación de reportes orientadas a usuarios finales (no programadores).
- Debe disponer de un lenguaje estándar de consulta a bases de datos (ej: SQL) que permita realizar consultas complejas y procesos de minería de datos independientemente de la interface de consulta de los usuarios finales.
- Debe contemplar permisos de acceso y seguridad que regulen y gestionen el acceso a datos, procesos, módulos y funciones según colecciones, bibliotecas, unidades administrativas, tipos de documentos, grupos de usuarios y usuarios

puntuales. Este tipo de parámetros debe ofrecer una interfaz de gestión para usuarios finales.

- Debe garantizar condiciones de trazabilidad de la totalidad de los procesos contemplados en el sistema garantizando condiciones de identificación unívoca de usuarios, tiempos, procesos y tareas.
- Debe realizar operaciones de ingreso, almacenamiento, edición, salida y visualización de datos a través del set de caracteres UNICODE/UTF-8.
- Debe disponer de clientes capaces de operar en condiciones de compatibilidad e integración (ejemplo copiar, pegar, cortar texto plano sin formato) con cualquier sistema operativo en las estaciones de trabajo.
- Debe garantizar condiciones de integridad de datos en procesos concurrentes en el contexto del sistema (ejemplo edición simultánea de registros, etc.).
- Debe permitir desarrollar extensiones funcionales capaces de articularse con plataformas de servicios de terceros (Web services, etc.).
- Debe establecer condiciones de identificación única para cada usuario a través de una única clave y nombre de usuario válida a través de todos los módulos y funciones del sistema.
- Debe permitir una navegación simple y lógica entre y dentro de los diferentes módulos.
- Debe permitir configurar valores por defecto y preferencias en cualquier módulo según grupos de usuarios, usuarios específicos y sesiones de trabajo.
- Debe disponer de ayudas contextuales integradas a la interfaz de gestión y documentación operativa completa.
- Debe disponer de capacidades para la internacionalización de la interface de consulta pública (OPAC).

Con base en las especificaciones generales dadas anteriormente el sistema contará con los módulos de:

- **Interfaz del usuario**

  Proveerá la documentación completa de ayuda en línea para todos los módulos y subsistemas.

- **Adquisición y control de existencias**

  Gestionará los procesos de compra a través de identificadores únicos (ejemplo número de proveedor, número de compra, número de pedido, número de resolución, número de expediente) persistentes e invocables a través de los distintos módulos y componentes del sistema.

  - Compra
  - Donación
  - Material gratuito
  - Canje
  - Depósito legal

- **Recepción, reclamo y registro de existencias**

  Catalogación

  - Debe permitir la gestión independiente de múltiples bibliotecas y colecciones.
  - Debe permitir la configuración de parámetros de acceso, servicios y seguridad independientes para cada biblioteca y colección.
  - Debe ofrecer condiciones de visibilidad y consulta de las relaciones bibliográficas existentes para cada ítem (ejemplares, adquisiciones, autoridades). Describir el alcance de la funcionalidad.
  - Debe permitir establecer permisos funcionales de alta, baja, modificación de datos, gestión de estados y acceso a datos según unidades administrativas,

colecciones específicas, tipos de materiales, perfiles de usuarios y usuarios puntuales.

- Debe contemplar condiciones de compatibilidad con formatos de intercambio MARC 21.
- Debe contemplar mecanismos de actualización y mantenimiento en la definición de campos y estándares de descripción de recursos.
- El módulo de catalogación deberá estar totalmente integrado con el módulo de gestión de autoridades, con el fin de poder efectuar automáticamente la validación de los campos de autoridad de los registros bibliográficos.

- **Gestión de autoridades**
  - Debe contemplar condiciones de compatibilidad con formatos de intercambio MARC 21 de autoridades.
  - Debe contemplar mecanismos de actualización y mantenimiento en la definición de campos y estándares de descripción de recursos aplicables a la gestión normalizada de autoridades.
  - El módulo de autoridades deberá estar articulado con todas las funcionalidades de gestión y descripción documental.
  - Debe permitir implementar mecanismos de exportación e importación de datos de autoridades.
  - Debe permitir realizar búsquedas delimitadas según campos y subcampos definidos en el esquema de descripción.
  - Debe permitir establecer y gestionar de manera consistente relaciones recíprocas (términos relacionados), asimétricas (jerarquía) y de equivalencia (términos preferidos) entre elementos.
  - Debe permitir la configuración de reglas complejas de composición de índices de autoridades (reglas sintácticas y algorítmicas), como por ejemplo; la integración de títulos de series en el índice de títulos, el tratamiento de títulos uniformes, etc.

- Debe permitir la parametrización completa de los esquemas de publicación Web de los datos de autoridades, tanto en términos de selección de campos publicables como así también en términos de formatos y esquemas de publicación (texto plano, XML, html, etc.).
- Debe contar con una sintaxis de búsqueda que permita buscar un término en cualquier campo de autoridades.
- Debe poder operar y brindar servicios a través de protocolos de consulta de registros de autoridades (Z39.50, SRU/SRW).

Funcionalidades de la interfaz de búsqueda

- Debe ofrecer URLs únicas, accesibles y persistentes a todas las páginas disponibles en la interfaz Web del OPAC.
- Debe ofrecer una interfaz de búsqueda estándar y una interfaz de búsqueda avanzada.
- La interfaz de búsqueda estándar debe contemplar búsquedas a través de todos los puntos de acceso definidos en el sistema y palabras clave. La interfaz de búsqueda avanzada debe permitir:
  - Combinar una expresión de búsqueda libre previa con otra expresión de búsqueda libre.
  - Combinar una expresión de búsqueda libre con los valores controlados previstos en los esquemas de descripción de recursos (autoridades, tipos de recursos, etc.).
  - Delimitar una expresión de búsqueda según intervalos temporales.
  - Delimitar una expresión de búsqueda según disponibilidad y situación de recursos.
  - Utilizar operadores de proximidad entre términos de una expresión de búsqueda.
- Debe soportar búsquedas insensibles a mayúsculas y minúsculas.
- Debe soportar búsquedas insensibles a caracteres acentuados o especiales.

- Debe soportar búsquedas según identificadores únicos (códigos, ISBN, números de control, número de registro, etc.).
- Debe permitir ordenar los resultados de búsquedas según criterios parametrizables por el usuario (alfabético de título, autor, tema, fecha, idioma, tipo de material, fecha de ingreso, disponibilidad, cantidad de consultas, relevancia, etc.).
- Debe permitir explorar los resultados de búsquedas a través de sucesivas paginaciones de resultados.
- Debe permitir cancelar búsquedas en proceso.

Presentación de contenidos

- Permitirá gestionar y parametrizar aspectos relativos a la disposición (layout), apariencia (look and feel), encabezamiento HTML (etiquetas meta del header HTML).
- Debe permitir administrar múltiples configuraciones de visualización según bibliotecas, unidades administrativas, colecciones, tipos de recursos y parámetros arbitrarios.
- Debe permitir establecer configuraciones de visualización en términos de:
    - Selección de campos y datos publicables
    - Tratamiento de campos y datos publicables
    - Orden de los campos y datos
    - Formateo de los datos (XML, HTML, PDF, texto plano, mrc, ISO 2709, etc.).
- Debe soportar para todas las operaciones de búsqueda, ordenamiento y presentación de datos caracteres UNICODE codificados a través de UTF-8.

Impresión y descarga de datos

- Debe soportar y prever medios de envío y distribución digital de registros.
- Debe permitir la visualización de todos los campos, incluyendo el volumen, ejemplar y el estatus de la información, para ser impreso, almacenado o reenviado electrónicamente.
- Debe permitir al usuario final especificar los elementos del registro a imprimir o guardar.
- La biblioteca debe poder configurar los formatos de salida y estructuras de datos que pueden ser exportados, almacenados o reenviados a través del OPAC.
- Debe permitir al usuario final seleccionar el formato y esquema de datos para imprimir o guardar un registro entre una lista de opciones configurable por la biblioteca.
- Debe permitir a los usuarios seleccionar y gestionar registros para su posterior impresión, almacenamiento o reenvío a través de una sesión de búsqueda.

Ayudas en contextos de búsqueda

- Debe proveer listas de opciones contextuales para búsquedas.
- Debe ofrecer opciones de ayuda y ejemplificación en todos los niveles e instancias de búsqueda y exploración.
- Debe prever mecanismos para la gestión y configuración local por parte de la biblioteca de los tutoriales y leyendas contextuales, mensajes de ayuda y de error.

Circulación

Se encargará de establecer los parámetros de control y gestión de plazos y condiciones de reserva, préstamo y renovación según patrones basados en tipos de materiales, colecciones, bibliotecas unidades administrativas, tipos de usuarios, etc.

Gestión de circulación y reservas

- Debe permitir identificar cada material y cada usuario tanto a partir de dispositivos de entrada ópticos (por ejemplo, código de barras) como manuales (por ejemplo, teclado).
- Debe permitir consultar los distintos límites que afectan a un recurso puntual en procesos de préstamo, renovación y reserva.
- Debe permitir consultar los distintos límites que afectan a un usuario puntual en procesos de préstamo, renovación y reserva.
- Debe permitir consultar los límites de préstamo, renovación y reserva para un recurso puntual en relación con un usuario puntual.
- Debe permitir bloquear préstamos a usuarios específicos por períodos de tiempo específicos.
- Debe permitir bloquear préstamos de recursos específicos por períodos de tiempo específicos.
- Debe permitir realizar procesos de renovación de préstamo y devolución masivos sobre conjuntos de materiales prestados a un usuario.
- Debe permitir modificar manualmente plazos de renovación, préstamo y reserva.
- Debe permitir emitir comprobantes impresos y electrónicos de las transacciones de préstamo, renovación, devolución y reserva. Describa niveles y modalidades de articulación con servidores de correo electrónico u otras plataformas de servicios (Web services, XML, etc.).
- Debe permitir establecer reportes corrientes según patrones basados en fechas, tipos de usuarios, tipos de materiales, biblioteca, colección y tipo de transacción (reserva, préstamo, renovación, devolución y reclamo).
- Debe permitir consultar las transacciones realizadas y solicitadas (préstamo, reserva, devolución, etc.) sobre un material desde cualquier módulo del sistema.

Préstamo interinstitucional

- Debe disponer de un módulo de préstamo interbibliotecario totalmente operacional.
- Debe permitir establecer reportes corrientes según patrones basados en fechas, tipos de usuarios, tipos de materiales, biblioteca, colección y tipo de transacción (solicitud, préstamo, renovación, devolución y reclamo).

Estadísticas de circulación

- Debe mantener estadísticas de uso para todos los servicios gestionados por el sistema.
- Debe mantener estadísticas específicas sobre procesos de préstamo, reservas, devoluciones y consultas de materiales.
- Debe mantener estadísticas capaces de considerar intervalos horarios de uso y circulación.

Generación de reportes

- Debe tener herramientas disponibles para la generación de informes ordinarios y extraordinarios y la personalización o adaptación de reportes.
- Describa los procedimientos disponibles para el almacenamiento y recuperación de parámetros de generación de informes y reportes.
- El sistema realizará reportes e informes programados de acuerdo a calendarios definibles.
- Las opciones disponibles para la salida de reportes e informes (impresión, en pantalla) y los formatos soportados (XML, excel, pdf, word, etc.).
- El sistema ofrecerá informes estándar.

Conversión e intercambio de datos (importación y exportación)

- Debe soportar procesos de intercambio de registros bibliográficos y de autoridades basados en estándares de servicios. Incluir descripción técnica, operativa, alcance y limitaciones relevadas.
- Debe soportar procesos de importación y exportación de registros bibliográficos y de autoridades (sin pérdida de datos) codificados en esquemas formales y estructurados de datos. Incluir descripción técnica, operativa, alcance y limitaciones relevadas.
- Soportar procesos de importación y exportación de registros de existencias (sin pérdida de datos) codificados en esquemas formales y estructurados de datos. Incluir descripción técnica, operativa, alcance y limitaciones relevadas.
- Debe permitir la parametrización por parte de la biblioteca, a través de una interfaz de usuario, de los campos a considerar como claves únicas para la evaluación de superposición de registros para cada proceso de importación.
- Debe contemplar procesos de auditoría e informes sobre procesos de importación y exportación. Especificar y describir los informes que desarrolla.
- Debe permitir realizar procesos de importación y exportación por lotes (batch) y en tiempo real.

Procesos de respaldo y recuperación de datos

- El sistema proveerá mecanismos de alerta y verificación sobre el grado de consistencia y éxito de los procesos de respaldo.
- Se requiere que el sistema disponga de capacidades para establecer procesos de respaldo de datos automáticos programados y procesos de respaldo eventuales.

Seguridad, autenticación y autorización

El sistema deberá ser capaz de realizar lo siguiente:

- Soporta la gestión de acceso y seguridad según niveles y funciones específicas.
- Permite a la biblioteca configurar derechos de acceso a funciones, bases de datos, módulos de gestión.
- Deberá soportar el acceso consistente a todos los espacios, servicios y funciones a través de un único punto de autenticación (login).
- Tendrá la posibilidad de establecer permisos de acceso específicos según usuarios y perfiles de usuarios a bibliotecas, colecciones, unidades administrativas, bibliotecas, grupos de registros, registros puntuales, etc.
- Deberá soportar la posibilidad de establecer permisos de acceso específicos según usuarios y perfiles de usuarios a campos y subcampos específicos.
- Contará con esquemas gerenciales de usuarios según unidades administrativas independientes.
- Permitirá la gestión de derechos de acceso según registros de autoridad, bibliográficos, de existencias, de adquisiciones, de control de publicaciones seriadas, de fondos y de circulación, por biblioteca, por colecciones, o por unidad de procesamiento o administrativa, etc.
- Proveerá de informes de auditoría de la actividad de los usuarios y de los administradores.
- Asegurará que múltiples usuarios sean bloqueados ante la edición concurrente de un registro dado.
- Dispondrá de capacidades para comunicarse con procesos de autenticación de usuarios independientes del sistema (APIs).
- Dispondrá de plantillas para la creación de perfiles de usuarios (definiciones básicas que pueden copiarse dentro de nuevos perfiles de usuario).

- Soportará modelos de control basados en la combinación de roles y atributos (ejemplo, un único usuario puede tener múltiples roles sin necesitar múltiples identificadores de usuario).
- Deberá soportar condiciones de compatibilidad con otros sistemas de autenticación y autorización.

Plan de migración ("conversión de datos")

- Deberá soportar las estrategias de migración de datos requeridas por la biblioteca a partir de las fuentes de datos descritas por la misma.
- Deberá contemplar esquemas de conversión y migración para los siguientes tipos de registros:
    - Bibliográficos.
    - Autoridades.
    - Ejemplares.
    - Proveedores.
    - Parámetros de carga y configuración.
    - Títulos y existencias en publicaciones seriadas.

Proceso de instalación

- Deberá contemplar un proceso de instalación y parametrización hasta lograr condiciones de implementación completa del sistema.
- Se describirá el orden esperado y el cronograma de los eventos en el proceso de instalación. Incluir un calendario, comenzando desde el mes 0 y continuando hasta la implementación completa.
- Se describirán los conocimientos y habilidades técnicas que se requieren para instalar e implementar el sistema.
- Se describirá un modelo de roles y responsabilidades sugeridas para el proveedor y la biblioteca en el proceso de instalación e implementación.

3.3 RECOPILACIÓN Y ANÁLISIS DE LA INFORMACIÓN

Los procesos actuales de la biblioteca proveen información para una administración óptima del material bibliográfico así como de los usuarios a los que proporciona el servicio.

Estos procesos pueden ser clasificados de manera muy general en tres grandes rubros.

- Administración del material bibliográfico.
- Administración de los usuarios de la biblioteca.
- Búsquedas y préstamo domiciliario a usuarios.

Administración del material bibliográfico

A su vez la administración del material bibliográfico se divide en dos ramas que son las nuevas adquisiciones y baja de material bibliográfico. Cuando un nuevo material es adquirido por la biblioteca pasa al área de nuevas adquisiciones donde se toma la información necesaria para clasificarlo.

- Autor.
- Título.
- Editorial.
- Observaciones especiales.
- No. de Registro (único)
- No. de Ejemplares.

Esta información es captura en un archivo de Excel, si el material ya se encuentra en existencia únicamente se agrega al consecutivo al número de ejemplares en existencia. En el caso que el material se haya registrado por primera vez en la biblioteca entonces se hace su clasificación y se crea su ficha bibliográfica que es agregada a los ficheros para la consulta de los usuarios.

En algunas ocasiones es necesario dar de baja el material bibliográfico, las razones pueden ser por extravío de los usuarios, mutilación o pérdida dentro de la biblioteca, para esos casos se busca el material en el archivo de Excel y se agrega en la última columna la leyenda “Baja” así como las observaciones de baja. Si el material es único entonces es removida de los ficheros su ficha bibliográfica correspondiente, en caso contrario no es removido la ficha de los ficheros. Figura 3.3.1.

Nombre del Campo	Tipo de Dato
No de Registro	Entero consecutivo
Autor	Cadena de caracteres
Titulo	Cadena de caracteres
Editorial	Cadena de caracteres
Observaciones	Cadena de caracteres
Número de ejemplares	Entero
Clasificación bibliográfica	Cadena de caracteres
Material Activo	Verdadero / Falso

Figura 3.3.1 Tabla de administración de material bibliográfico

Administración de los usuarios de la biblioteca

Al igual que el material, los usuarios deberán estar debidamente registrados en los archivos que el personal bibliotecario maneja para éste fin. Cuando un nuevo usuario es registrado se le solicita la siguiente información.

- Nombre del usuario.
- Dirección.
- No. de cuenta.
- Teléfono particular.
- Teléfono celular.
- Correo electrónico.
- Número de usuario. (único).

Los bibliotecarios solicitan mínimo una referencia para que en el caso de pérdida del material o incidentes mayores, la referencia pueda responder por el material prestado. De la misma manera de cómo guardan la información de los libros, se guarda la información de los usuarios, en archivos de Excel. Figura 3.3.2.

Nombre del Campo	Tipo de Dato
Número de usuario	Entero consecutivo
Nombre del Usuario	Cadena de caracteres
RFC	Cadena de caracteres
Clave de usuario	Cadena de caracteres
Fecha de alta	Fecha
Fecha de último acceso	Fecha
Rol de usuario	Cadena de caracteres
Teléfono particular	Cadena de caracteres
Teléfono celular	Cadena de caracteres
Correo electrónico	Cadena de caracteres
Nombre de la referencia 1	Cadena de caracteres
Dirección de la referencia 1	Cadena de caracteres
Nombre de la referencia 2	Cadena de caracteres
Dirección de la referencia 2	Cadena de caracteres
Nombre de la referencia 3	Cadena de caracteres
Dirección de la referencia 3	Cadena de caracteres

Figura 3.3.2 Tabla de administración de usuarios

Búsquedas y préstamo domiciliario a usuarios

Cuando el material es dado de alta se crean las fichas bibliográficas, los usuarios directamente consultan ahí la información de los recursos bibliográficos, los trabajadores tienen esa misma información pero capturada en los archivos mencionados de Excel. Cuando los usuarios hacen uso del préstamo domiciliario, los bibliotecarios buscan en los archivos de Excel, si la persona que ha sido registrada previamente, entonces es un usuario de la biblioteca, si es un usuario válido entonces le otorgan el servicio.

Para prestarle el servicio el personal bibliotecario llena un archivo de Excel con las especificaciones del material prestado, día de devolución y usuario que solicita el préstamo. Al final de la jornada laboral todos los archivos de Excel se concatenan en uno solo para tener el vaciado de todos los libros que se prestaron en el día. En la parte posterior del libro se sella con la fecha de entrega, con el sello el libro puede entrar y salir de la biblioteca sin problema alguno. Tabla 3.3.3.

Nombre del Campo	Tipo de Dato
Número de control de préstamo	Entero consecutivo
Clave del libro para préstamo	Cadena de caracteres
Número de usuario	Cadena de caracteres
Nombre del usuario	Cadena de caracteres
Número del empleado	Fecha
Nombre del empleado	Fecha
Fecha de préstamo	Fecha
Fecha de devolución	Fecha

Figura 3.3.3 Tabla de administración de préstamos

3.4 IDENTIFICACIÓN DE LOS POSIBLES MÓDULOS DEL SISTEMA

En el sistema de control bibliotecario de usuario se definirán los siguientes módulos para su funcionamiento operacional:

- Módulo de Circulación.
- Módulo de Catalogación.
- Módulo de Directorio.
- Módulo de Pagos.
- Módulo de Consulta.
- Módulo de Administración.

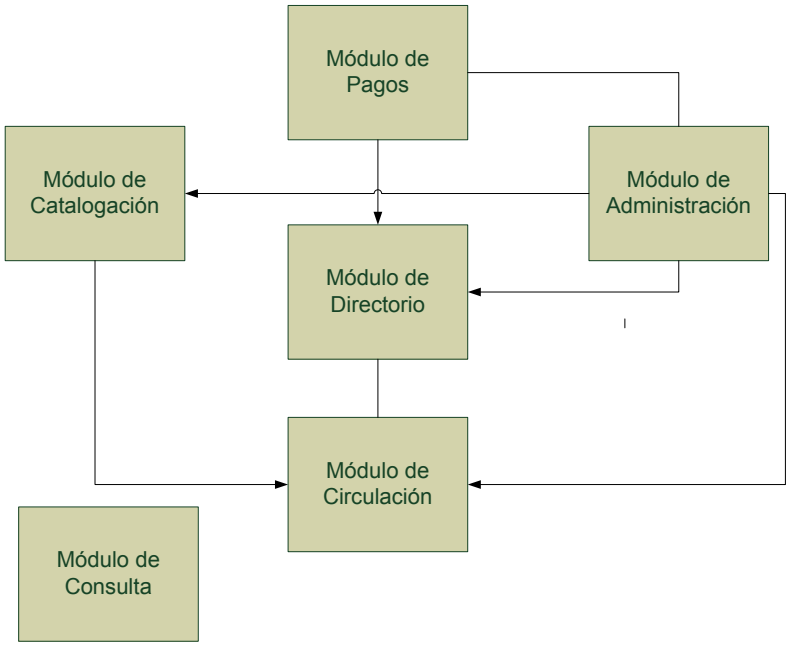


Figura 3.4.1 Diagrama de bloques

Como se muestra en la figura 3.4.1 se puede observar que los módulos estarán interactuando entre sí para poder cumplir el objetivo del funcionamiento operacional del sistema.

Módulo de circulación

En el módulo de circulación se llevarán a cabo todos los movimientos del material catalogado en la figura 3.4.2 se puede observar el diagrama de bloques del módulo de circulación.

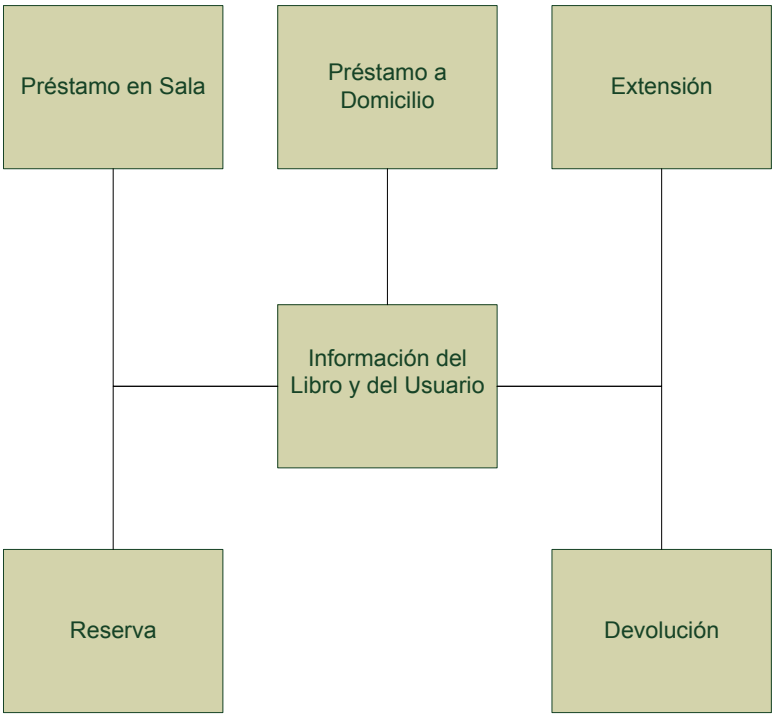


Figura 3.4.2 Diagrama módulo de circulación

Préstamo en sala

El préstamo en sala se refiere a cuando un usuario de la biblioteca pide un libro sólo para ser consultado dentro de las instalaciones de la biblioteca, el libro no puede salir de la biblioteca.

Préstamo a domicilio

Se refiere a cuando un usuario de la biblioteca tiene todos los requisitos necesarios que pide el sistema de bibliotecas, estos son: que tenga la credencial vigente, no tenga adeudo de libros vencidos, no tenga adeudos de multas por conceptos como multas por libros no devueltos a tiempo y colegiaturas. El préstamo a domicilio puede ser de 3 días a 8 días dependiendo de cuantos días se le hayan otorgado a ese usuario en el módulo de directorio.

Extensión

Se refiere a cuando un usuario de la biblioteca que tiene un libro prestado puede pedir una extensión del préstamo, esto dependerá de tenga esos privilegios y de que no se haya vencido el tiempo del préstamo original.

Reserva

Se refiere a cuando un usuario de la biblioteca quiere un libro pero este libro esta prestado a otro usuario, entonces lo que puede hacer es reservar el libro para una fecha posterior a cuando el libro sea devuelto esto para asegurar que nadie más pueda hacer un préstamo del mismo.

Devolución

Es cuando un usuario que tiene un libro prestado lo devuelve en la biblioteca, si el libro es entregado en tiempo no hay ningún otro tramite pero si es entregado después de la fecha que lo tenía prestado se le tendrá que levantar una multa por los días que se haya pasado de la fecha que lo tenía prestado.

Información del libro y del usuario

Se refiera a si el libro se encuentra disponible para poder ser prestado, si no lo tiene alguien prestado a domicilio o si no está reservado por otro usuario.

La información del usuario es: si el usuario tiene una credencial vigente, si no tiene adeudos (multas), si no ha superado el límite de libros prestados.

Módulo de catalogación

Seguirá unas normas internacionales que deben conocerse para evitar organizar la biblioteca con criterios personales que impidan ser continuados por otras personas. Sin embargo es necesario simplificarlas y adaptarlas a las características de las bibliotecas.

Lo importante es hacer dinámica la biblioteca y las normas sólo facilitan la tarea de organización y circulación de los materiales. La informatización permite hacer la catalogación con un mínimo conocimiento de dichas reglas y permite ahorrar tiempo en las tareas bibliotecarias.

Si en alguna biblioteca se quisiera seguir con el sistema de fichas, bastará con trabajar un primer nivel de catalogación con los siguientes elementos:

- Signatura topográfica.
- Apellidos y nombre del autor.
- Título de la obra.
- Lugar de publicación.
- Editorial.
- Año de publicación.
- Descriptores (del contenido de la obra).
- Número de registro.

La catalogación de los documentos sigue las normas ISBD (International Standard Bibliographic Description) o Norma Internacional de Descripción Bibliográfica, que

establecen cuáles son las áreas de descripción, las fuentes de información y prescribe la puntuación de separación de áreas y elementos.

La mayoría de los datos que son necesarios para catalogar un libro se obtendrán de la portada. De la portada se extraerá la información del autor, título, editorial, dibujantes, traductores, etc.

La contraportada contiene el resto de datos: colección, título original, edición, ISBN, depósito legal, etc.

Para seleccionar los descriptores o temas de los que trata un documento, los equipos docentes tendrán a su disposición el lenguaje documental ordenado alfabéticamente y por temas.

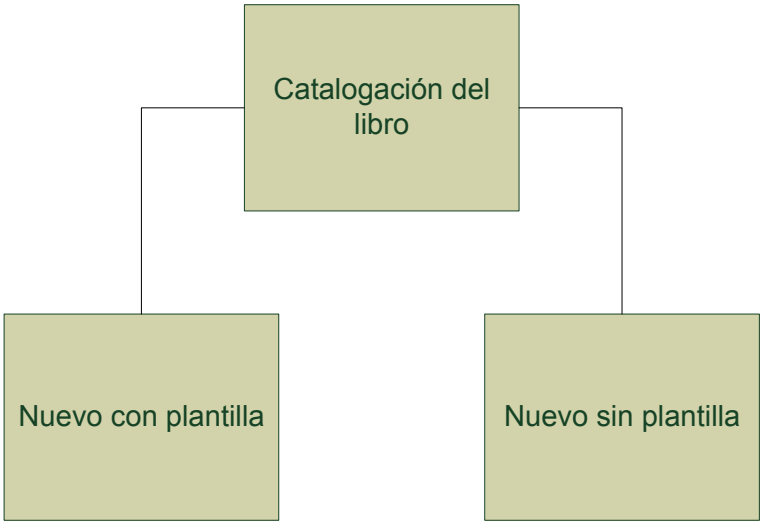


Figura 3.4.3 Diagrama módulo de catalogación

Como se muestra en la figura 3.4.3 se observa que los bloques de nuevo sin plantilla y nuevo con plantilla se refieren a si se hace una catalogación nueva se puede usar una plantilla predefinida o no.

Módulo de Directorio

En este módulo se guardará toda la información del usuario de la biblioteca estos datos son:

- Información domicilio.
  - Calle.
  - Numero.
  - Colonia.
  - Delegación.
  - Estado.
  - País.
  - Teléfono.
  - Email.
- Información personal.
  - Nombre completo.
  - Sexo.
  - Estado civil.
  - RFC.
  - CURP.
  - Fecha de nacimiento.
  - Nacionalidad.
- Información de la empresa.
  - RFC de la empresa.
  - Giro.
  - Dirección.
  - Teléfonos.
  - Email de la empresa.
- Información de expedientes.
  - Tipo de expediente.
  - Descripción.

- Notas.

- Información de la biblioteca.
  - Tipo de lector.
  - Vigencia.
  - Situación.
  - Material prestado.
  - Material reservado.
  - Material autorizado.
- Información del alumno.
  - Nivel.
  - Grado.
  - Grupo.
- Información general.
  - Datos de facturación.

En la figura 3.4.4 se muestran los bloques de información que recibirá el módulo de directorio.

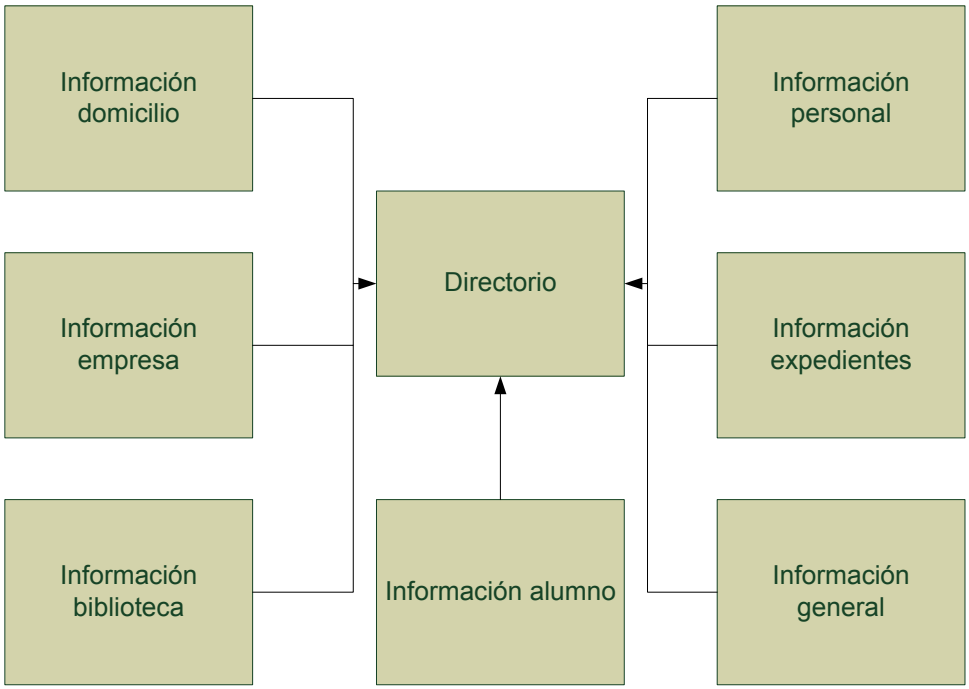


Figura 3.4.4 Diagrama módulo de directorio

Módulo de pagos

Se llevará el control de los pagos y adeudos que tenga un usuario los cuales pueden ser multas o recargos por no regresar un ejemplar a tiempo, u otros servicios que la biblioteca provea, como pueden ser el cobro de copias, costo de la credencial de servicios y otros conceptos que se puedan utilizar.

Módulo de consulta

Se refiere a búsquedas especializadas muy fáciles de llevar a cabo y muy rápidas en su respuesta, existen varios criterios de búsqueda lo que facilita encontrar lo que el usuario esta buscando.

Con éste módulo se busca que los usuarios de la biblioteca no requieran ayuda de personal propio de la institución para realizar sus búsquedas ya que se pretende que el sistema sea sencillo y amigable para cualquier usuario.

Módulo de administración

Se pretende utilizar para guardar los parámetros de configuración del sistema así como también catálogos para todos los demás módulos como se muestra en la figura 3.4.5. Los catálogos pueden ser para el módulo de directorio, módulo de circulación y módulo de pagos.

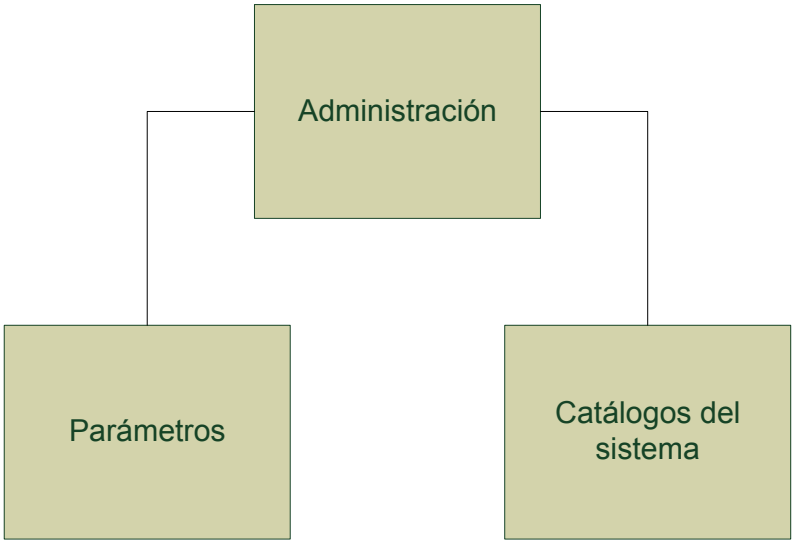


Figura 3.4.5 Diagrama módulo de administración

3.5 COMPARACIÓN DE LAS HERRAMIENTAS DE SOLUCIÓN.

Microsoft VB o Delphi

Una de las primeras de las primeras cuestiones fue determinar el lenguaje de programación que se usaría para desarrollar la aplicación. Basándonos en que el usuario podía proporcionarnos tanto Visual Basic como Delphi para el desarrollo de dicha aplicación, presentamos una serie de comparaciones para determinar cuál es el lenguaje de programación más conveniente, ¿VB o Delphi?

Realmente no existe un lenguaje ideal, si bien los programadores con experiencia en Delphi tienen conflictos si se cambian a VB, los programadores de VB no poseen ese tipo de dificultades al pasar a Delphi.

Si se analizan ambos sistemas desde el punto de vista económico, ésto es; formación, desarrollo, etc., VB es mucho más rápido de aprender que Delphi. Por lo regular una persona en menos de un mes con conocimientos generales-básicos de programación, está preparada para empezar a desarrollar aplicaciones escritas en VB, sin embargo, para la misma persona en Delphi, el tiempo se alarga a 3 veces. La formación por otra parte es más cara en Delphi que en VB y la complejidad es directamente proporcional.

Sin embargo, la posibilidad de tener un lenguaje POO u orientado a objetos, hacen de Delphi una herramienta muy potente.

Por el contrario, VB es precompilado y no posee un archivo ejecutable puro (se recomienda crear un archivo .exe con VB y abrirlo con un editor de texto, el código está ahí). Por esta razón necesita librerías DLL para poder ejecutarse.

Delphi al contrario es un lenguaje compilado y esto redunda en su rapidez y fiabilidad

¿Qué es entonces lo que busca VB con las DLL principales para poder ejecutar un programa?

VB hace uso de las DLLs para la justificación de espacio en las aplicaciones desarrolladas ya que Delphi requiere que se incluyan las DLL en el ejecutable por lo que ocupa más tamaño. En relación al espacio es correcto, sin embargo, tiene por desventaja que las DLL y ActiveX suelen modificarse con frecuencia en cuanto a los Service Pack o parches se refiere, por lo que la problemática es aún mayor que el simple tamaño de un archivo ejecutable.

Microsoft se ha acercado un poco a la idea de la conjunción de un único ejecutable en su código mediante el P-Code, esto no se acerca a la realidad. En precio, VB es mucho más económico que Delphi y eso es un Handicap muy importante para un particular o una empresa.

La potencia de Delphi respecto a VB parece estar justificada en cuanto a que Delphi es Pascal y acepta ensamblador para ejecutar algunas instrucciones, sin embargo, VB posee la posibilidad de ejecutar otras instrucciones potentes sin usar la complejidad de ensamblador, si bien, no es tan potente.

Los términos explicados en estas líneas, son algunos de los argumentos que defienden a ambos lenguajes como idóneos, los cuales continúan ligando una de las batallas más largas entre lenguajes de programación diferente.

El hecho de ser los dos lenguajes de programación visuales más extendidos y utilizados en el mundo, hacen a su vez, tener adeptos y enemigos acérrimos, si bien es conveniente saber y conocer que ambos lenguajes son igualmente válidos para una programación seria y profesional.

Curva de aprendizaje

Visual Basic es fácil de aprender y de utilizar, no solamente porque el lenguaje de programación no es OOP y es fácil de codificar (al final viene de BASIC), sino también porque el IDE es simple y cómodo de usar, y los objetos de base de datos que vienen

con Visual Basic son más fáciles de utilizar que sus contrapartes de Delphi, aunque claro que no son tan potentes.

Visual Basic hace muchas cosas por el programador. Por ejemplo, los objetos cuentan referencias y esto significa que si creamos un objeto asignándolo a una variable local, el objeto será liberado automáticamente cuando la función o el procedimiento finalice (a menos que lo asignamos a una variable no local). Visual Basic tiene un sistema de administración sofisticado de memoria y utiliza un "colector de basura" (garbage colector) así que es rápido liberando memoria.

En cuanto al acceso a bases de datos en Visual Basic es más simple, se usa un solo componente que abre el Recordset o DataSet dependiendo la versión, ofrece interfaz visual de navegador y se enlaza con los controles de datos; en Delphi hay que usar tres componentes para ello. La gran ventaja de los Recordsets de Visual Basic sobre los Datasets de Delphi es que los primeros manejan consultas actualizables automáticamente: se puede tener una consulta de dos tablas y que sea "viva", mientras que en Delphi tenemos que utilizar actualizaciones cacheadas y un componente UpdateSQL con las respectivas consultas SQL para insertar, actualizar o eliminar un registro. Además, cuando uno actúa contra un servidor de base de datos en Delphi tiene que utilizar un componente Transaction, mientras que esto no es necesario en Visual Basic. Visto desde la perspectiva de un programador Visual Basic, el acceso a datos de Delphi resulta ser demasiada molestia cuando se lo compara con los Recordsets y el Control de Datos de Visual Basic que simplifican notoriamente esta cuestión.

Rendimiento y tamaño del código

El único ítem donde se observa que Visual Basic tiene un buen rendimiento es en el acceso a datos, y quizá en los informes, porque la capa de acceso a datos y la generación de reportes no están hechas con Visual Basic, pero en lo que respecta

estrictamente a generación de código, a pesar del hecho que Visual Basic ahora produce código nativo en vez de pseudocódigo (P-Code), todavía está muy lejos de alcanzar una velocidad comparable a Delphi, por más optimizaciones que uno le active. Esto sucede porque Delphi tiene un compilador optimizante y contiene opciones de compilación para optimizar la velocidad de su código. Además, si uno no está contento con el funcionamiento de una rutina en Delphi, puede afinar el código para optimizar hasta el último nanosegundo, hasta el punto de incluir instrucciones en lenguaje ensamblador. En lo que hace a rendimiento de código, Delphi sea ideal para tareas que requieren programación pesada.

Extensibilidad

Visual Basic y Delphi son extensibles. Ambos IDEs soportan "add-ons" (agregados) para añadir asistentes y otras extensiones que permiten la posibilidad de utilizar objetos visuales y no visuales aparte de los que vienen originalmente con la herramienta de desarrollo.

Objetos ActiveX

En Visual Basic uno puede utilizar objetos ActiveX. Vienen en archivos .OCX que son como DLLs que tienen funciones especiales para exponer objetos con sus propiedades, métodos y eventos. El acceso a propiedades y métodos de objetos ActiveX es lento porque se acceden con IDs. Las propiedades de cadena de los objetos ActiveX son UNICODE, mientras que las aplicaciones trabajan normalmente con cadenas ANSI, agregando otra sobrecarga (la conversión UNICODE-ANSI o ANSI-UNICODE) al acceder a propiedades de cadena.

Los objetos ActiveX proporcionan encapsulación, pero no herencia. La herencia se puede simular declarando los mismos métodos y propiedades de la "clase base" e implementándolos como llamadas a los métodos y las propiedades respectivos de la "clase base". La "clase derivada" es por consiguiente una envoltura de la "clase base".

No sólo que esto hace el código fuente largo y difícil de mantener, sino que también hace la invocación de propiedades y de métodos del nuevo objeto más de dos veces de lenta que el objeto original.

Disponibilidad de bibliotecas y objetos de terceros

Hay muchos objetos ActiveX y DLLs en la red que se pueden utilizar con Visual Basic, pero la mayoría es de paga y no vienen con código fuente, ni siquiera después que uno paga sus licencias, y cuando traen código fuente, por lo general éste se encuentra en otro lenguaje (generalmente Visual C++ o Delphi), y esto significa que si uno desea modificar estos fuentes necesita 1) saber algo de ese lenguaje, y 2) tener la herramienta de desarrollo respectiva, y no creemos que valga la pena comprar una licencia tan sólo para ocasionalmente hacer modificaciones menores a objetos y bibliotecas.

Las perspectivas en Delphi son mucho mejores, porque además de ActiveX y de DLLs, uno puede encontrar muchos componentes VCL en la red, siendo freeware la mayoría de ellos, muchos de ellos vienen con código fuente, y la mayoría de los que son de paga vienen generalmente con el código fuente después de que uno lo registra o por una tarifa adicional. ¿Y en qué lenguaje está escrito el código fuente? Una muy pequeña minoría están en C++ Builder, pero la enorme mayoría están en Delphi, por supuesto, así que uno no tiene que aprender un nuevo lenguaje o utilizar una herramienta de desarrollo diferente si desea modificarlos.

Obsolescencia

Casi todos los desarrolladores en Visual Basic por lo regular están programando en Visual Basic 6, o bien se encuentran en el cambio a C#. Con Delphi sucede algo totalmente diferente. Mucha gente todavía está programando en Delphi 6 o 7, y quizás la minoría tiene Delphi 7, cuando Delphi 2009 acaba de salir a la luz pero las actualizaciones son demasiado caras.

Lo que sucede con Visual Basic es que los programadores Visual Basic se actualizan rápidamente tan pronto como hay una nueva versión disponible. Delphi 6 es tan poderoso que todavía hoy no es obsoleto, y esto explica por qué muchos programadores en Delphi 6 no ven la necesidad de pagar por una actualización cuando ya tienen lo que necesitan. La extensibilidad tiene también mucho que ver con esto.

Los programadores Visual Basic necesitan actualizarse demasiado rápido. Delphi es tan poderoso, funcional y extensible que sus versiones no se hacen obsoletas fácilmente: a la larga, Delphi es más barato.

Poder de programación y funcionalidad.

Delphi permite:

1. Escribir aplicaciones, bibliotecas, objetos ActiveX y componentes VCL de calidad comercial, rápidos y pequeños.
2. Crear reportes por código (o modificar existentes, por ejemplo para cambiar o agregar una columna).
3. En general, todo lo que se puede hacer visualmente (en tiempo de diseño) se puede hacer programáticamente (en tiempo de ejecución).
4. Escribir fácilmente aplicaciones multi-hilos.
5. Ver los datos de las tablas y consultas en tiempo de diseño.
6. Llamar a funciones de la API como si fueran funciones incorporadas.
7. Introducir código en lenguaje ensamblador para mejor desempeño donde sea necesario.

8. Aprovechar los montones de componentes VCL disponibles para darle a sus aplicaciones un "look and feel" distintivo y para ofrecer una funcionalidad mejorada.

9. Derivar fácilmente componentes VCL existentes para añadirles funcionalidad.

Delphi ofrece a programadores un poder y una funcionalidad que deja a Visual Basic y sus limitaciones atrás.

Comparativo de manejadores de bases de datos libres

En un inicio las empresas que se peleaban los primeros lugares en cuestiones de manejadores de bases de datos DBMS (Database Management System) y además dominaban el mercado eran IBM, Oracle y Microsoft. En la actualidad existen herramientas de código abierto (Open Source) que van tomando más madurez y son mejor vistos por la industria de software gracias a la madurez que han alcanzado. Además un aspecto muy importante que ha hecho que herramientas de código abierto sean ahora tan populares es el económico. Por lo anterior se trató de hacer una evaluación con aplicaciones de código abierto para determinar cuál sería la más óptima.

Las herramientas que se evaluaron son las siguientes:

- Firebird 1.5.2
- Ingres r3 3.0.1
- MaxDB 7.5.0.23
- MySQL 4.1.10
- Postgres 8.0.1

Licencia

El primer punto importante sobre una comparación y evaluación de herramientas es la licencia, ya que el decir de código abierto no necesariamente significa que los DBMS pueden ser usados libremente en aplicaciones comerciales.

Firebird 1.5.2	El módulo original fue liberado por Inprise con la licencia InterBase Public Licence (IPL)[12]. Los nuevos módulos agregados a Firebird se encuentran con la licencia Initial Developer's Public License (IDPL). Ambas versiones son modificaciones de la licencia Mozilla Public License v.1.1 Firebird puede ser distribuido completamente incluso para aplicaciones comerciales.
Ingres r3 3.0.1	Ingres se encuentra disponible bajo la licencia Computer Associates Trusted Open Source. Ingres puede ser distribuido completamente incluso para aplicaciones comerciales.
MaxDB 7.5.0.23	MaxDB se encuentra disponible bajo la licencia de MySQL Dual Licensing Model. • Commercial Licence • GNU General Public License Información sobre el precio comercial se encuentra disponible en la página del producto.
MySQL 4.1.10	MySQL se encuentra disponible bajo la licencia MySQL Dual Licensing Model. • Commercial License • GNU General Public License Información sobre el precio comercial se encuentra disponible en la página del producto.
PostgreSQL 8.0.1	PostgreSQL se encuentra liberado bajo la licencia BSD license. PostgreSQL puede ser distribuido completamente incluso para aplicaciones comerciales.

[12] La versión actual de InterBase Public License se encuentra disponible en el sitio Web de Borland

Plataformas

Las plataformas más importantes por los DBMS son mostradas en la siguiente tabla.

	Sistema Operativo	Arquitectura
Firebird 1.5.2	Microsoft Windows 95 / 98 / ME	X86_32
	Microsoft Windows 2000	X86_32
	Microsoft Windows XP	X86_32
	Microsoft Windows Server 2003	X86_32
	Linux	X86_32
	FreeBSD	X86_32
	Sun Solaris	X86_32
	Sun Solaris	Sparc, X86_32
	HP – UX	PA – RISC
	Mac OS X	PPC_32
	Sistema Operativo	**Arquitectura**
Ingres r3 3.0.1	Microsoft Windows 2000	X86_32
	Microsoft Windows XP	X86_32
	Microsoft Windows Server 2003	X86_32
	Linux	X86_32
	Sun Solaris	Sparc, 32-bit y 64-bit
	HP – UX (aún en versión	PA-RISC, 32-bit y 64-bit

	beta)	
MaxDB 7.5.0.23	**Sistema Operativo**	**Arquitectura**
	Microsoft Windows 2000	X86_32
	Microsoft Windows XP	X86_32
	Microsoft Windows Server 2003	X86_32, X86_64, IA64
	Linux	X86_32, X86_64, IA64, PPC_64
	Sun Solaris	Sparc, 64-bit
	HP – UX	PA-RISC, IA64
	HP Tru64 UNIX	Alpha
	IBM AIX	PPC_64
MySQL 4.1.10	**Sistema Operativo**	**Arquitectura**
	Microsoft Windows 95 / 98 / ME	X86_32
	Microsoft Windows 2000	X86_32
	Microsoft Windows XP	X86_32
	Microsft Windows Server 2003	X86_32
	Linux	X86_32, X86_64, IA64, Alpha, S/390
	FreeBSD	X86_32
	Sun Solaris	X86_32, Sparc 32-bit y 64-bit
	HP – UX	PA-RISC, IA64
	Mac OS X	PPC_32
	IBM AIX	RS6000
	QNX	X86_32
	Novell Netware	X86_32

	Sistema Operativo	Arquitectura
PostgreSQL 8.0.1	Microsoft Windows 2000	X86_32
	Microsoft Windows XP	X86_32
	Miscrosoft Windows 2003 Server	X86_32
	Linux	X86_32, X86_64, ARM, IA64, PA-RISC, MIPS, PPC_32. PPC_64, Sparc, S/390
	FreeBSD	X86_32
	Sun Solaris	Sparc, X86_32
	HP – UX	PA-RISC, IA64
	HP Tru64 UNIX	Alpha
	Mac OS X	PPC_32
	IBM AIX	PPC_32, RS6000

Límites de la Base de Datos

El administrador y el desarrollador debe ser conciente de los límites de la base de datos en la que está desarrollando las aplicaciones, los límites de la base incluyen, el tamaño máximo de la base, máximo número de archivos por base, máximo número de tablas en la base, etc., el término ilimitado debe de usarse con precauciones, ya que todos los límites que se mencionan tienen como límite el tamaño disponible en el disco.

Firebird 1.5.2	• Número máximo de tablas: 32767. • Tamaño máximo de la base: Teóricamente limitada a 7 TB. • Máximo número de archivos por base: Teóricamente 65536 (2^{16}) incluyendo archivos tipo shadow. • Tamaño máximo de páginas: 16384. • Máximo buffers de caché: 65536.
Ingres r3 3.0.1	• Número máximo de tablas: 67108863

	• Tamaño máximo de la base: Ilimitada • Máximo número de archivos por base: Ilimitada. • Tamaño máximo de páginas: 65536.
MaxDB 7.5.0.23	• Número máximo de tablas: Ilimitado • Tamaño máximo de la base: 32 TB (con 8 kb de tamaño de página).
MySQL 4.1.10	• Número máximo de tablas: Ilimitado • Tamaño máximo de la base: Ilimitado • Máximo número de archivos por base: Ilimitada.
PostgreSQL 8.0.1	• Número máximo de tablas: Ilimitado • Tamaño máximo de la base: Ilimitado. • Máximo número de archivos por base: No determinado.

Límites de querys complejos

A continuación se muestra una tabla comparativa entre las diferentes herramientas en las que se muestra la potencia que tienen en manejar querys complejos entre los cuales se encuentran el número máximo de sentencias “IN” en una lista, número máximo de tablas dentro de sentencias “JOIN”, número máximo de operadores “AND/OR” dentro de sentencias “WHERE” o “SELECT”.

Firebird 1.5.2	• Tamaño máximo de sentencias SQL: 64 KB. • Número máximo de miembros “IN” dentro de una lista: 1499. • Número máximo de “JOIN” en tablas: 255. • Máximo número de operadores lógicos en sentencia “WHERE”: AND: 255; OR: Sin límite encontrado
Ingres r3 3.0.1	• Tamaño máximo de sentencias SQL: Configurable (hasta ≥ 64 KB). • Número máximo de miembros “IN” dentro de una lista: Sin límite encontrado.

	• Número máximo de “JOIN” en tablas: 126. Máximo número de operadores lógicos en sentencia “WHERE”: AND: 1023; OR: 1617.
MaxDB 7.5.0.23	• Tamaño máximo de sentencias SQL: ≥ 16 (Valor por default 64 KB, especificado en una variable del sistema). • Número máximo de miembros “IN” dentro de una lista: 2041, usando el valor por default en el parámetro PACKET_SIZE. • Número máximo de “JOIN” en tablas: 64. • Máximo número de operadores lógicos en sentencia “WHERE”: AND, OR: 511, usando el valor por default en el parámetro PACKET_SIZE.
MySQL 4.1.10	• Tamaño máximo de sentencias SQL: 1 GB, 16 MB por default, el tamaño depende del parámetro max_allowed_packet. • Número máximo de miembros “IN” dentro de una lista: sin límite encontrado. • Número máximo de “JOIN” en tablas: 61. • Máximo número de operadores lógicos en sentencia “WHERE”: AND, OR: sin límite encontrado.
PostgreSQL 8.0.1	• Tamaño máximo de sentencias SQL: Sin información disponible. • Número máximo de miembros “IN” dentro de una lista: 10917, usando el parámetro por default max_stack_depth. • Número máximo de “JOIN” en tablas: Sin límite encontrado. • Máximo número de operadores lógicos en sentencia “WHERE”: AND, OR: Sin límite encontrado.

Interface en la base de datos

A continuación se realizarán las comparaciones de diferentes aspectos de lo DBMS, primero se mostrará si el producto maneja el estándar SQL, además también el número de lenguajes que puede soportar así como si es que incluye APIs nativas de lenguajes de alto nivel.

Estándar SQL

Se mostrará si la herramienta soporta el estándar SQL-92.

Firebird 1.5.2	El lenguaje de Firebird SQL se mantiene fiel al estándar SQL-92, sin embargo Firebird introdujo un número considerado de características de acuerdo con el estándar SQL-99. A pesar de que FirebirdSQL sigue el estándar existen pequeñas diferencias.
Ingres r3 3.0.1	Es estándar SQL-92 es soportado totalmente, sin embargo el estándar SQL-92 puede ser deshabilitada para manejar versiones anteriores de Ingres.
MaxDB 7.5.0.23	MaxDB puede operar en diferentes modos de SQL: - INTERNAL: Definición del sistema interno de la base. - ANSI: Estándar ANSI de acuerdo con X3.135-1992. - DB2: Definición de DB2 Versión 4. - ORACLE: Definición de ORACLE7.
MySQL 4.1.10	Existen bastantes diferencias entre el estándar SQL y MySQL.
PostgreSQL 8.0.1	PostgreSQL soporta un subconjunto tanto del estándar SQL-92 así como del estándar SQL'99. En la documentación propia de PostgreSQL menciona que soporta y que no soporta.

Interface de lenguajes de programación

La disponibilidad de los lenguajes de programación es un requerimiento muy importante para las bases de datos ya que los desarrolladores acceden a la base de datos por medio de diferentes lenguajes que son integrados en el DBMS. En general los DBMS

tienen disponible alguna característica a través de una API nativa. Los drivers de alto nivel tienen disponible para el cliente una API que es fácil de usar dentro de una objeto modelo o alguna reimplementación entre el protocolo de comunicación del cliente y el servidor.

Firebird 1.5.2	La interface "nativa" de Firebird es a través de bibliotecas en C, funciones y estructuras de parámetros son expuestas en la API. El archivo cabecera de C se llama ibase.h, el cual se encuentra dentro del directorio /include dentro de la carpeta de Firebird. Éste archivo de cabecera se usa cuando se escriben programas que usan la biblioteca, además también sirve como referencia cuando se desarrolla interface para bibliotecas en otros lenguajes. Existen muchos drives de "alto – nivel" disponibles para Firebird como son: JDBC, ODBC, .NET Provider, y Framewors orientados a objetos para desarrolladores en C++ llamado "IBPP", componentes para Delphi, Kylixx y Borland C++ además se encuentran disponibles drivers para PHP, Python y Perl.
Ingres r3 3.0.1	Existen drivers disponibles para JDBC, ODBC y .NET. Bibliotecas OpenAPI también pueden ser usadas como una alternativa, el cual se encuentran disponibles los siguientes lenguajes; C, Ada, Cobo, Fortran, Pascal y PL1. El soporte para cada uno de estos bibliotecas ESQL depende de la plataforma usada.
MaxDB 7.5.0.23	MaxDB se encuentra abierta a muchos lenguajes de programación vía interface, la cual separa la base de datos y la lógica de aplicación para una mejor escalabilidad. Para el desarrollo de aplicaciones MaxDB soporta el MaxDB ODBC Driver, SQL DataBase Connectivity (SQLDBC), JDBC,

	módulos de Perl y Python y extensiones de PHP.
MySQL 4.1.10	Existen APIs disponibles para C, PHP, Perl, C++, Python, Tcl. Los drivers disponibles son; JDBC, ODBC y .NET
PostgreSQL 8.0.1	Diferentes tipos de interface se encuentran disponibles, como son; ODBC, JDBC y ECPG, interface embebida de SQL en C, y bibliotecas de Tcp igualmente se encuentran disponibles.

Conclusiones

Visual Basic es adecuado para simples aplicaciones de interfaz de usuario, típicamente aplicaciones de gestión (facturación, inventarios, nóminas, etc.). Su facilidad de uso lo hace la opción correcta para los programadores principiantes. Delphi es aún mejor para aplicaciones de interfaz de usuario pues la disponibilidad de componentes VCL permite que uno desarrolle interface de calidad superior tanto en términos de funcionalidad como de presentación, pero no es tan fácil de utilizar para los principiantes, que son el grupo mayoritario y para quienes prevalece la facilidad sobre la calidad del trabajo, velocidad de ejecución, tamaño del código, uso de los recursos del sistema, tiempo de desarrollo, costos o cualquier otro factor.

Se considera que solamente es posible convencer a un ejecutivo para que elija Delphi en vez de Visual Basic cuando hay requisitos específicos que lo justifican, por ejemplo si la compañía no puede tener el último hardware, no pueden comprar las actualizaciones a la última versión de Visual Basic, no puede permitirse pagar soluciones cada vez que se topan con una limitación en Visual Basic, necesitan una GUI (Interfaz de Usuario Gráfica) de mejor calidad (más funcional y visualmente más atractiva) o cuando el personal de Visual Basic no puede entregar las aplicaciones en término (esto sucede normalmente cuando se están tropezando con alguna limitación, puesto que usualmente se complica el código y entonces se hace difícil de mantener) o

cuando se requieren menores tiempos de desarrollo para aplicaciones que no son tan simples.

Delphi es más difícil de usar que Visual Basic al principio, no hay duda de ello, pero permite hacerlo todo. Como alguien dijo, "Delphi es más compatible con Windows que Visual Basic". Aquellos programadores que cargan en su equipaje algún conocimiento de un lenguaje orientado a objetos (como C++, Turbo Pascal 5.5+ o FreePascal) no tienen problemas para aprender Delphi. Son habitualmente aquellos que han tenido una base de educación formal en programación (universidades, politécnicos, etc.) o aprendieron por sí mismos. El mercado de Delphi no está limitado a las aplicaciones de interfaz de usuario: Delphi se usa también para el back-end (servidores, bibliotecas, objetos ActiveX, etc.), un mercado fuera del alcance de Visual Basic.

Visual Basic hace muchas cosas por los programadores, así que muchos que lo usan están muy cómodos con él (probablemente Delphi les parezca demasiado arcano), y si no se han encontrado con las limitaciones de Visual Basic, ¿por qué cambiar? Visual Basic es la herramienta correcta para ellos.

Delphi es una herramienta profesional para programadores profesionales, mientras Visual Basic es para aquellos que se conforman con lo poco que la herramienta les brinda ya sea porque no necesitan más, no tienen suficiente preparación o experiencia en la programación en Windows o en programación orientada a objetos, o porque no demandan demasiado de la herramienta de programación, o no quieren o no les gusta programar, o no tienen tiempo para aprender, o no saben inglés, o porque con Visual Basic es más fácil conseguir un trabajo porque las empresas prefieren contratar gente que llene ese perfil. A pesar de ello, debe decirse que Visual Basic es una gran herramienta y que ha puesto la programación al alcance de muchos profesionales que vienen de carreras fuera de la programación (Analistas de Sistemas de Información, Ingenieros en Electrónica, etc.).

Las corporaciones prefieren tener muchos candidatos disponibles cuando necesitan cubrir un puesto de trabajo, que tener pocos candidatos para cubrir una posición que requiera más calificaciones, y por esta razón prefieren VB sobre Delphi.
En nuestro caso elegimos Delphi porque el manejador de base de datos tiene un mejor soporte de lenguaje para Delphi que para Visual Basic.

En comparativas para el manejador de la base de datos se eligió Firebird ya que es una base totalmente madura que puede proporcionar los servicios y en la documentación es la que soporta mejor el lenguaje Delphi, pues contiene una API totalmente nativa la cual las otras herramientas no la contenían por ello nos inclinamos a elegir Firebird además de que no exige un equipo potente para manejar la base.

La mejor herramienta para el trabajo es la que mejor se adapta a sus requerimientos, limitaciones y expectativas.

CAPÍTULO 4

4.1 ELECCIÓN DE LA METODOLOGÍA DE DESARROLLO

La metodología seleccionada para el proyecto es UML ya que nos permitirá visualizar, especificar, construir y documentar sistemas desde la perspectiva orientada a objetos, de una forma sencilla, clara y concisa.

Una de las ventajas por las que se seleccionó dicha metodología UML es que prescribe una notación estándar y semánticas, esenciales para el modelado de un sistema orientado a objetos. Previamente, un diseño orientado a objetos podría haber sido modelado con cualquiera de la docena de metodologías populares, causando a los revisores tener que aprender las semánticas y notaciones de la metodología empleada antes que intentar entender el diseño en sí. Con UML, diferentes diseñadores pueden modelar muchos sistemas ya que maneja estándares los cuales son fáciles de entender, por ello cualquier modelo realizado con UML es fácil de entender y desarrollar.

UML nos permite dividir cada proyecto en un número de diagramas que representan las distintas vistas del proyecto y juntos representan la arquitectura del mismo por lo que permite describir un sistema en diferentes niveles de abstracción.

Los diagramas de clases de UML se pueden usar para modelar la base de datos relacional en la que un sistema esté basado, aunque sin embargo los diagramas tradicionales de modelado capturan más información sobre la base de datos relacional y son más adecuados para modelarla.

Como extensión de UML, el diagrama de clases puede ser referenciado en un diagrama de entidad relación (ER) el cual relaciona entidades que pueden ser modeladas basadas en atributos clave.

Con los diagramas de caso de uso se da el punto de entrada para analizar los requisitos del sistema, y el problema que es necesario solucionar.

Las siguientes secciones presentan una vista más detallada al modelado con UML.

- Organización del sistema por medio de paquetes.
- Modelando casos de uso y usándolos para averiguar los requisitos del sistema.
- Modelado con diagramas de secuencia y diagramas de colaboración.
- Modelado de comportamiento con diagramas de actividad y de estado.
- Modelado de componentes de software, distribución e implementación.
- Es posible extender UML con el diseño de bases de datos relacionales.

Una de las tareas clave para modelar un sistema de software de grandes dimensiones es dividirlo primero en áreas manejables. Aunque estas áreas se llaman dominios, categorías o subsistemas, la idea es la misma; dividir el sistema en áreas que tengan competencias parecidas.

UML introduce la noción de un paquete como el ítem universal para agrupar elementos, permitiendo a los modeladores subdividir y categorizar sistemas. Los paquetes pueden ser usados en cualquier nivel, desde el nivel más alto, donde son usados para subdividir el sistema en dominios, hasta el nivel más bajo, donde son usados para agrupar casos de uso individuales, clases, o componentes.

Las ventajas de usar UML:

- Facilita el diseño y la documentación.
- Código reutilizable.
- Descubrimiento de fallas.
- Ahorro de tiempo en el desarrollo del software.
- Son más fáciles de realizar las modificaciones.
- Existe una comunicación fluida entre programadores.

El intercambio de información de diseño e ideas usando la notación UML puede ser realizado en los medios que siempre han sido populares: pizarrones, cuadernos y trozos de papel por nombrar algunos. Pero UML puede ser usado con alguna herramienta de modelado, la cual puede ser usada para capturar, guardar, rechazar, integrar automáticamente información y diseño de documentación, por ejemplo Clear Case.

Una característica que beneficia a los modeladores es que UML hace más fácil escoger una herramienta de modelado. Anteriormente el modelador tenía que seleccionar una notación de metodología, y después estaba limitado a seleccionar una herramienta que la soportara. Con UML como estándar, la elección de notación ya se ha hecho para el modelador. Y con todas las herramientas de modelado soportando UML, el modelador puede seleccionar la herramienta basada en las áreas claves de funcionalidad soportadas que permiten resolver los problemas y documentar las soluciones.

Como una buena herramienta de modelado ofrece todas las armas necesarias para hacer eficientemente varios trabajos. Esto es:

- Soporte para toda la notación y semántica de UML.
- Soporte para una cantidad considerable de técnicas de modelado y diagramas para complementar UML - incluyendo tarjetas CRC, modelado de datos, diagramas de flujo, y diseño de pantallas de usuario.

- Posibilidad de reutilizar información obtenida por otras técnicas todavía usadas, como modelado tradicional de procesos.
- Facilitar la captura de información en un repositorio subyacente - permitiendo la reutilización entre diagramas.
- Posibilidad de personalizar las propiedades de definición de elementos subyacentes de modelos UML.
- Permitir a varios equipos de analistas trabajar en los mismos datos a la vez.
- Posibilidad de capturar los requisitos, asociarlos con elementos de modelado que los satisfagan y localizar cómo han sido satisfechos los requisitos en cada uno de los pasos del desarrollo.
- Posibilitar la creación de informes y documentación personalizados en tus diseños, y la salida de estos informes en varios formatos, incluyendo HTML para la distribución en la Internet o Intranet local.
- Posibilidad para generar y 'revertir' código (por ejemplo C++, Java, etc.) para facilitar el análisis y diseño 'iterativo' y volver a usar código o librerías de clase existentes, o igualmente para documentar el código.

4.2 DIAGRAMACIÓN

4.2.1 DIAGRAMA DE CONTEXTO

El diagrama de contexto modela la relación del sistema con los elementos externos, pues dichos elementos son los que forman el contexto del sistema. Los pasos para realizar un diagrama de contexto son:

- Identificar los actores que interactúan con el sistema.
- Organizar a los actores.
- Especificar las vías de comunicación con el sistema.

En la siguiente figura se muestra el diagrama de contexto del sistema. Figura 4.2.1.1

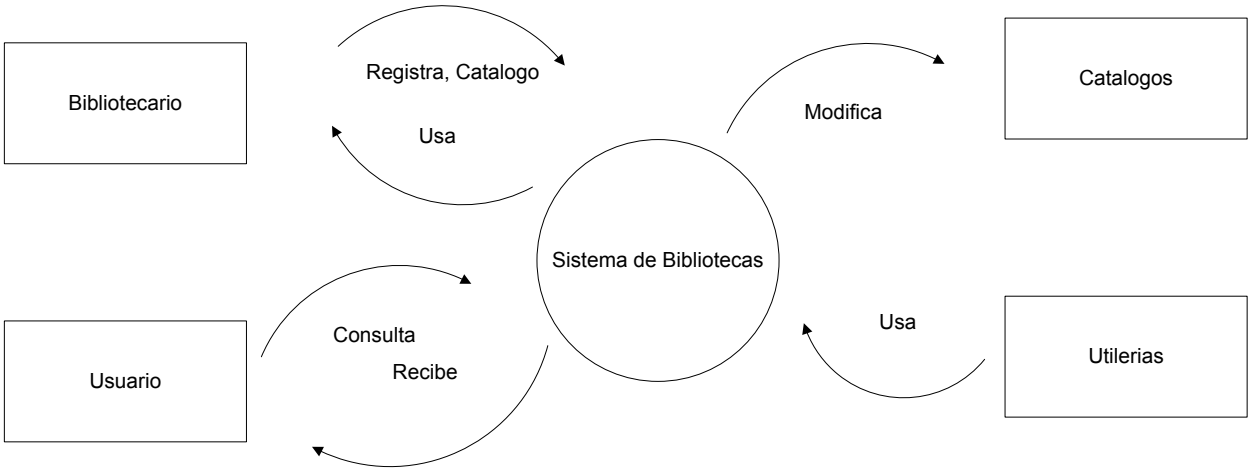


Figura 4.2.1.1 Diagrama de contexto

4.2.2 DIAGRAMA DE CASOS DE USO

El diagrama de casos de uso muestra los casos de uso, actores y sus relaciones, además muestra quién puede hacer qué y las relaciones existentes entre las acciones (casos de uso). El diagrama de casos de uso es muy importante para modelar y organizar el comportamiento del sistema.

A continuación se muestran los diagramas de casos de uso del sistema. Figura 4.2.2.1.

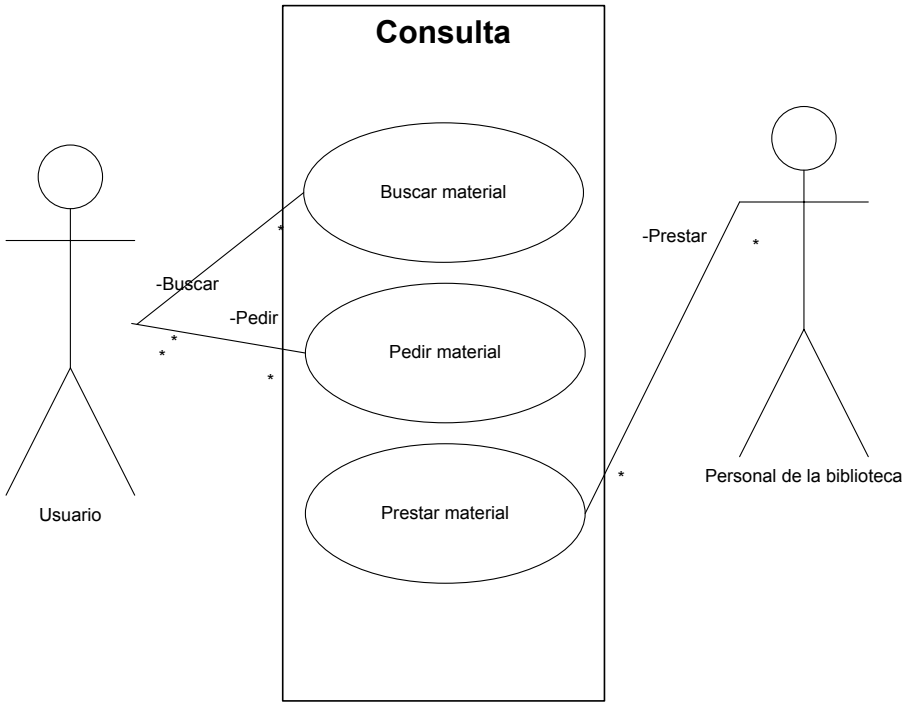


Figura 4.2.2.1 Caso de uso del módulo de consulta

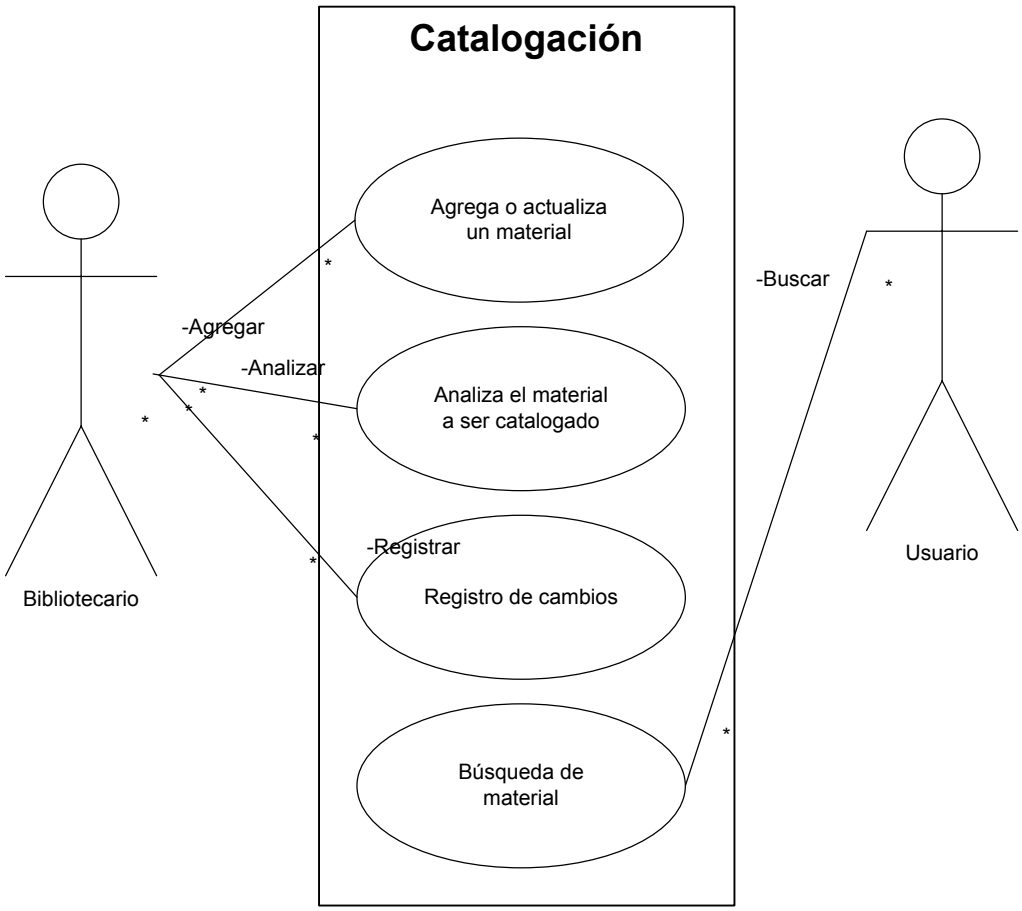


Figura 4.2.2.2 Caso de uso del módulo de catalogación

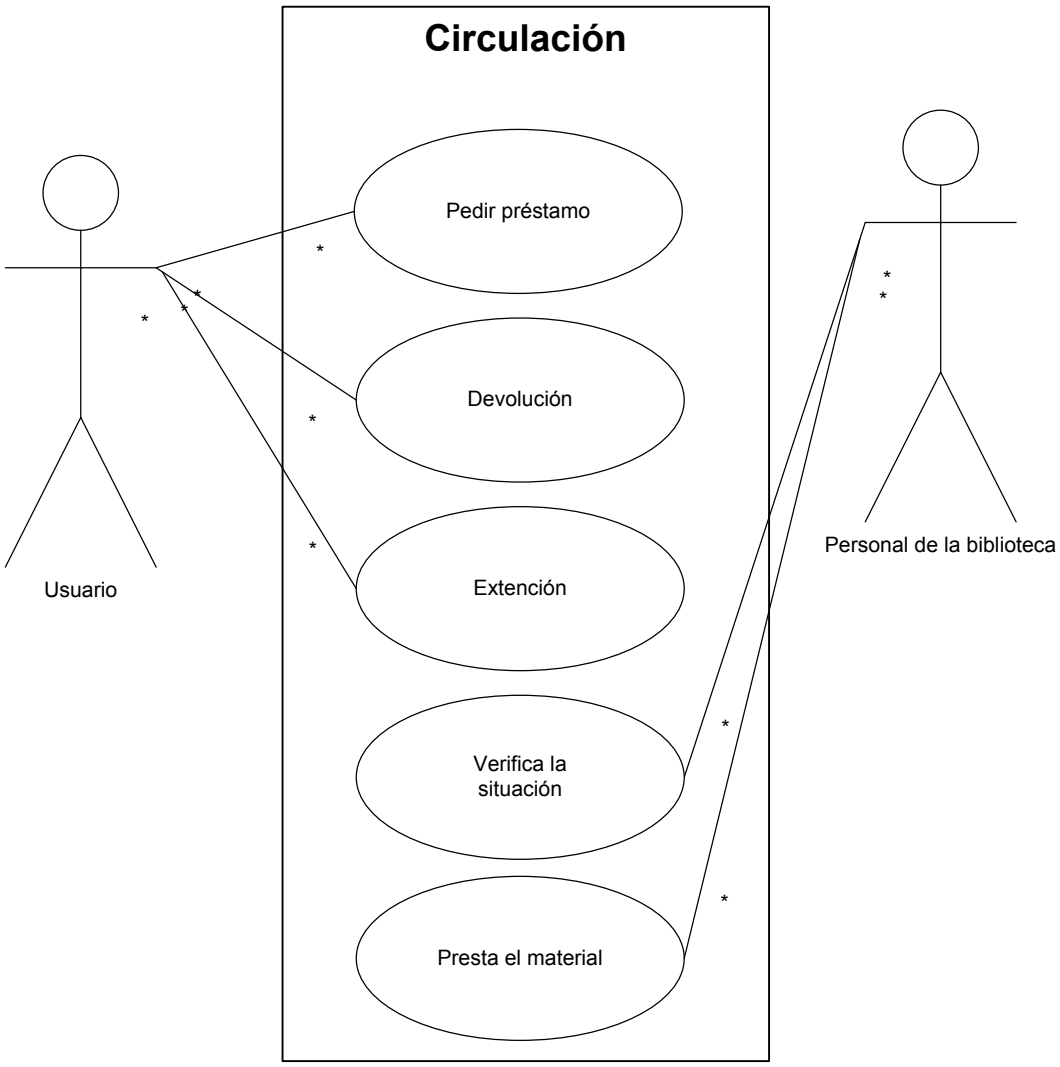


Figura 4.2.2.3 Caso de uso del módulo de circulación

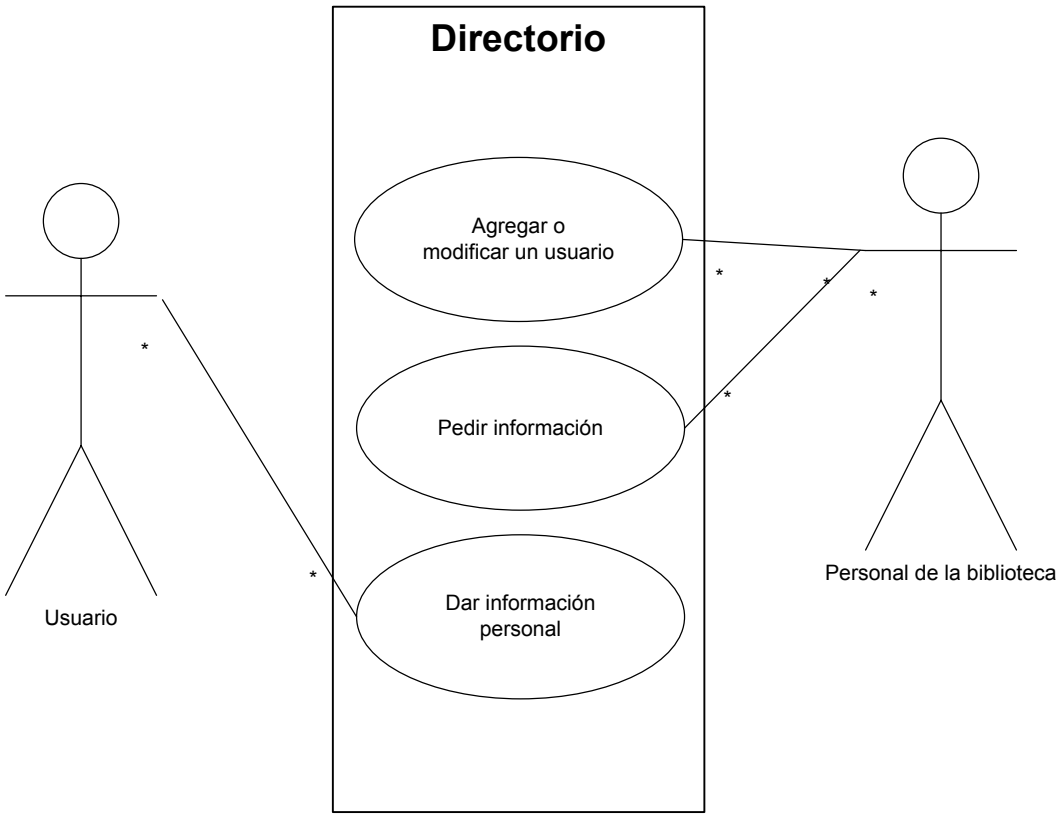


Figura 4.2.2.4 Caso de uso del módulo de directorio

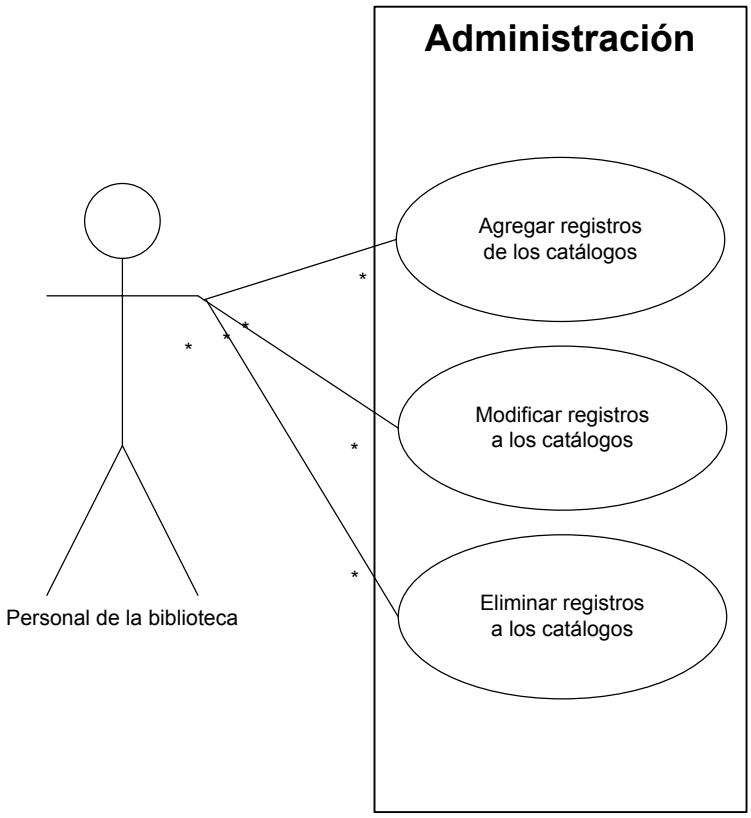


Figura 4.2.2.5 Caso de uso del módulo de administración

4.2.3 DICCIONARIO DE DATOS

El diccionario de datos es un listado organizado de todos los datos que pertenecen a un sistema.

Como su nombre lo sugiere, estos elementos se centran alrededor de los datos y la forma en que están estructurados para satisfacer los requerimientos de los usuarios y las necesidades de la organización. Los elementos más importantes que contiene son:

- Las características lógicas de los sitios donde se almacenan los datos del sistema, incluyendo nombre, descripción, alias, contenido y organización.
- Los procesos donde se emplean los datos y los sitios donde se necesita el acceso inmediato a la información.
- Datos acerca de los datos que comúnmente son llamados metadatos.

Razones para su utilización:

- Para manejar los detalles en el sistema, ya que tienen enormes cantidades de datos, aun en los sistemas más chicos.
- Manejar todos los detalles ya que los sistemas sufren cambios continuos. Auxilia en el análisis y diseño del software.
- Los analistas mas organizados usan el diccionario de datos específicamente para el análisis y diseño de software.
- Proporcionan asistencia para asegurar significados comunes para los elementos y actividades del sistema, registrando detalles adicionales relacionados con el flujo de datos, de tal manera que todo pueda localizarse con rapidez.
- Documentar las características del sistema, incluyendo partes o componentes así como los aspectos que los distinguen, procesos y frecuencia de estos.

- Facilitar el análisis de los detalles con la finalidad de evaluar las características y determinar donde efectuar cambios.
- Determina si son necesarias nuevas características o si están en orden los cambios de cualquier tipo.
- Localizar errores y omisiones en el sistema, detectar dificultades, y presentarlas en un informe. Aún en los manuales, se revelan errores.
- El contenido de un registro del diccionario, ver figura 4.2.3.1, es un conjunto de columnas que contienen:
    - Tipo de dato de la columna.
    - Acrónimo.
    - Longitud del tipo de dato.
    - Llave que pueden ser primaria o foránea.
    - Acepta nulos.
    - Tablas con las que se asocia.
    - Tipo de datos.
    - Descripción.

Nobre de Tabla:					
Nombre de Columna	Tipo de dato de Columna	Opción de Columna Vacia	Columna es PK	Columna es FK	Descripción

Figura 4.2.3.1Contenido de un registro del diccionario

A continuación se muestran los diccionarios de datos con las descripciones de las tablas:

- Figura 4.2.3.2 Descripción de D_478_Canal_Distribucion.
- Figura 4.2.3.3 Descripción de Empresa.

- Figura 4.2.3.32 Tipo_Concepto.
- Figura 4.2.3.33 D_233_Producto.
- Figura 4.2.3.34 D_38_Tipo_Tarjeta.

Nobre de Tabla:	D_478_Canal_Distribucion				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
MedioEn	DECIMAL(6)	NOT NULL	Yes	No	MedioEn
Descripcion_Larga	VARCHAR(256)	NULL	No		Descripción Larga
Descripcion_Corta	VARCHAR(128)				Descripción Corta
Descripcion_Pantalla					Descripción Pantalla
Abrev	VARCHAR(10)				Abreviacióniación
Fecha_Alta	DATE				Fecha Alta
Fecha_U_Mod					Fecha U Mod
Fecha_Baja					Fecha Baja

Figura 4.2.3.2 Descripción de D_478_Canal_Distribucion

Nobre de Tabla:	Empresa				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Empresa	DECIMAL(3)	NOT NULL	Yes		Identificador Empresa
descripcion	VARCHAR(256)		No		Descripción
descripcion_Corta	VARCHAR(128)				Descripción Corta
siglas	VARCHAR(8)				siglas
estatus	DECIMAL(1)	NULL			estatus

Figura 4.2.3.3 Descripción de Empresa

Nobre de Tabla:	Unidad_Medida				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Unidad_Medida	DECIMAL(3)	NOT NULL	Yes		Identificador Unidad Medida
descripcion	VARCHAR(256)		No		Descripción
descripcion_Corta	VARCHAR(128)				Descripción Corta
siglas	VARCHAR(8)				siglas
estatus	DECIMAL(1)	NULL			estatus

Figura 4.2.3.4 Unidad Medida

Nobre de Tabla:	Tipo_Promocion				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Tipo_Promocion	DECIMAL(3)	NOT NULL	Yes		Identificador Tipo Promocion
descripcion	VARCHAR(256)		No		Descripción
descripcion_Corta	VARCHAR(128)				Descripción Corta
condiciones_Evaluacion	DECIMAL(1)	NULL			condiciones Evaluacion
condiciones_Iniciales					condiciones Iniciales
condiciones_por_Archivo					condiciones por Archivo
siglas	VARCHAR(8)	NOT NULL			siglas
estatus	DECIMAL(1)	NULL			estatus

Figura 4.2.3.5 Tipo Promoción

Nobre de Tabla:	Empresa_xTipo_Promocion				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Empresa	DECIMAL(3)	NOT NULL	Yes	Yes	Identificador Empresa
id_Tipo_Promocion					Identificador Tipo Promocion

Figura 4.2.3.6 Empresa_xTipo_Promocion

Nobre de Tabla:	s41_Empleado				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
idEmpleado	DECIMAL(16)			No	idEmpleado
idDepartamento				Yes	idDepartamento
id_Empresa	DECIMAL(3)				Identificador Empresa
nombre	VARCHAR(128)		No	No	nombre
apellidoPaterno					apellidoPaterno
apellidoMaterno					apellidoMaterno
idRol	DECIMAL(16)			Yes	idRol

Figura 4.2.3.7 s41_Empleado

Nobre de Tabla:	s41_Departamento				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
idDepartamento			Yes	No	idDepartamento
id_Empresa	DECIMAL(3)			Yes	Identificador Empresa
descripcion	VARCHAR(128)		No	No	Descripción
nombreCorto	VARCHAR(7)				Nombre Corto
descripcionImpresion	VARCHAR(127)				DescripciónImpresion

Figura 4.2.3.8 s41_Departamento

Nobre de Tabla:	s41_Rol				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
idRol	DECIMAL(16)		Yes		idRol
descripcion	VARCHAR(128)		No		Descripción
nombreCorto	VARCHAR(7)				Nombre Corto
descripcionImpresion	VARCHAR(127)				DescripciónImpresion

Figura 4.2.3.9 s41_Rol

Nobre de Tabla:	Promocion				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Promocion	DECIMAL(6)		Yes		Identificador Promocion
clave_Promocion	VARCHAR(12)	NULL	No		clave Promocion
nombre	VARCHAR(256)	NOT NULL			nombre
inicio_Promocion	DATE				inicio Promocion
fin_Promocion					fin Promocion
evaluacion	INTEGER				evaluacion
aplicacion					aplicacion
mes_Inicia_Aplicacion		NULL			mes Inicia Aplicacion
dirigida_A	CHAR(1)	NOT NULL			dirigida A
no_Maximo_Suscriptores	INTEGER	NULL			no Maximo Suscriptores
condicion_Inicial	VARCHAR(2048)				condicion Inicial
cancelada	BINARY	NOT NULL			cancelada
MedioEn	DECIMAL(6)	NULL		Yes	Medio
mecanismo_Evaluacion	DECIMAL(1)			No	mecanismo Evaluacion
id_Tipo_Promocion	DECIMAL(3)	NOT NULL		Yes	Identificador Tipo Promocion
estatus	DECIMAL(1)	NULL		No	estatus

Figura 4.2.3.10 Promoción

Nobre de Tabla:	Condicion				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Condicion	DECIMAL(2)	NOT NULL	Yes		Identificador Condicion
id_Promocion	DECIMAL(6)			Yes	Identificador Promocion
descripcion	VARCHAR(256)	NULL	No	No	Descripción
sql_sentencia	VARCHAR(2048)				sql sentencia
sql_verbalizado					sql verbalizado

Figura 4.2.3.11 Condición

Nobre de Tabla:	Beneficio				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Beneficio	DECIMAL(6)	NOT NULL	Yes		Identificador Beneficio
CveTran				Yes	CveTran
GpoAfin					GpoAfin
Product	DECIMAL(9)				Product
descripcion	VARCHAR(256)		No	No	Descripción
descripcion_Corta	VARCHAR(128)	NULL			Descripción Corta
id_Tipo_Saldo	DECIMAL(3)			Yes	Identificador Tipo Saldo
id_Unidad_Medida					Identificador Unidad Medida
id_Canal_Servicio					Identificador Canal Servicio
id_Programa_Lealtad					Identificador Programa Lealtad
CveFlag					CveFlag

Figura 4.2.3.12 Condición

Nobre de Tabla:	Auditoria

Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
idAuditoria	DECIMAL(16)	NOT NULL	Yes	No	idAuditoria
tabla	VARCHAR(128)		No		tabla
idAccion	DECIMAL(16)			Yes	idAccion
idEmpleado					idEmpleado
idDepartamento					idDepartamento
fecha	DATE			No	fecha
id_Empresa	DECIMAL(3)			Yes	Identificador Empresa

Figura 4.2.3.13 Auditoria

Nobre de Tabla:	Accion				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
idAccion	NUMBER(16)		Yes	No	idAccion
descripcion	VARCHAR(256)		No		Descripción
descripcionCorta	VARCHAR(128)				Descripción Corta
siglas	VARCHAR(8)				siglas

Figura 4.2.3.14 Acción

Nobre de Tabla:	s41_Tipo_Promocion_Rol				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
idRol	DECIMAL(16)		Yes	Yes	idRol
id_Tipo_Promocion	DECIMAL(3)				Identificador Tipo Promocion

Figura 4.2.3.15 s41_Tipo_Promocion_Rol

Nobre de Tabla:	Canal_Servicio				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Canal_Servicio				No	Identificador Canal Servicio
descripcion	VARCHAR(256)		No		Descripción
descripcion_Corta	VARCHAR(128)				Descripción Corta
siglas	VARCHAR(8)				siglas
estatus	DECIMAL(1)	NULL			estatus

Figura 4.2.3.16 Canal_Servicio

Nobre de Tabla:	Concepto_xProducto				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
CveTran	DECIMAL(6)	NOT NULL	Yes	Yes	CveTran
GpoAfin					GpoAfin
Product	DECIMAL(9)				Product

Figura 4.2.3.17 Concepto_xProducto

Nobre de Tabla:	Precio_Especial				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Condicion	DECIMAL(2)				Identificador Condicion
id_Promocion	DECIMAL(6)				Identificador Promocion
id_Beneficio					Identificador Beneficio
CveTran					CveTran
GpoAfin					GpoAfin
Product	DECIMAL(9)				Product
valor	DECIMAL(14,2)		No	No	valor
numero_Veces	DECIMAL(3)				numero Veces

Figura 4.2.3.18 Precio_Especial

Nobre de Tabla:	Programa_Lealtad				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Programa_Lealtad			Yes		Identificador Programa Lealtad
descripcion	VARCHAR(256)		No		Descripción
descripcion_Corta	VARCHAR(128)				Descripción Corta
siglas	VARCHAR(8)				siglas
estatus	DECIMAL(1)	NULL			estatus

Figura 4.2.3.19 Programa_Lealtad

Nobre de Tabla:	Tipo_Saldo				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Tipo_Saldo	DECIMAL(3)	NOT NULL	Yes		Identificador Tipo Saldo
descripcion	VARCHAR(256)		No		Descripción
descripcion_Corta	VARCHAR(128)				Descripción Corta
siglas	VARCHAR(8)				siglas
estatus	DECIMAL(1)	NULL			estatus

Figura 4.2.3.20 Tipo_Saldo

Nobre de Tabla:	Transaccion_Aplicada				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
numero_Producto	VARCHAR(18)	NOT NULL	Yes	Yes	numero Producto
fecha_Corte_Ciclico	DATE				fecha Corte Ciclico
id_Condicion	DECIMAL(2)				Identificador Condicion
id_Promocion	DECIMAL(6)				Identificador Promocion
id_Beneficio					Identificador Beneficio
CveTran					CveTran
GpoAfin					GpoAfin
Product	DECIMAL(9)				Product
id_Transaccion	DECIMAL(12)		No	No	Identificador Transaccion
fecha	DATE				fecha
nombre	VARCHAR(128)	NULL			nombre
monto	DECIMAL(12,2)	NOT NULL			monto

Figura 4.2.3.21 Transaccion_Aplicada

Nobre de Tabla:	Mot_NA_Beneficio				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Motivo_NA_Beneficio	DECIMAL(3)		Yes		Identificador Motivo NA Beneficio
descripcion	VARCHAR(256)		No		Descripción
descripcion_Corta	VARCHAR(128)				Descripción Corta
siglas	VARCHAR(8)				siglas
estatus	DECIMAL(1)	NULL			estatus

Figura 4.2.3.22 Mot_NA_Beneficio

Nobre de Tabla:	Suscripcion				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Promocion	DECIMAL(6)	NOT NULL	Yes	Yes	Identificador Promocion
numero_Producto	VARCHAR(18)				numero Producto
fecha_Suscripcion	DATE		No	No	fecha Suscripcion
fecha_Inicio_Evaluacion		NULL			fecha Inicio Evaluacion
fecha_Inicio_Aplicacion					fecha Inicio Aplicacion
id_Motivo_NA_Beneficio	DECIMAL(3)			Yes	Identificador Motivo NA Beneficio
estatus	DECIMAL(1)	NOT NULL		No	estatus

Figura 4.2.3.23 Suscripción

Nobre de Tabla:	Mot_No_Suscripcion				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Motivo_NA_Beneficio	DECIMAL(3)		Yes		Identificador Motivo NA Beneficio
descripcion	VARCHAR(256)		No		Descripción
descripcion_Corta	VARCHAR(128)				Descripción Corta
siglas	VARCHAR(8)				siglas
estatus	DECIMAL(1)	NULL			estatus

Figura 4.2.3.24 Mot_No_Suscripcion

Nobre de Tabla:	Cliente_Producto				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
numero_Producto	VARCHAR(18)	NOT NULL	Yes		numero Producto
fecha_Corte	DATE		No		fecha Corte
sol_id_Entidad_Federativa	VARCHAR(128)	NULL			sol Identificador Entidad Federativa
sol_id_Sucursal					sol Identificador Sucursal
sol_id_Canal_Venta					sol Identificador Canal Venta
sol_id_Empresa_Promotora					sol Identificador Empresa Promotora
sol_Score	DECIMAL(1)				sol Score
GpoAfin	DECIMAL(6)			Yes	GpoAfin
Product	DECIMAL(9)				Product

Figura 4.2.3.25 Cliente_Producto

Nobre de Tabla:	Comportamiento_Producto				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
numero_Producto	VARCHAR(18)	NOT NULL	Yes		numero Producto
fecha_Corte_Ciclico	DATE			No	fecha Corte Ciclico
saldo_corte	DECIMAL(12,2)		No		saldo corte
saldo_promedio					saldo promedio
compras					compras
disposiciones_efectivo					disposiciones efectivo

Figura 4.2.3.26 Comportamiento_Producto

Nobre de Tabla:	Tipo_Producto				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Tipo_Producto	DECIMAL(3)		Yes		Identificador Tipo Producto
id_Tipo_Producto_Prent			No	Yes	Identificador Tipo Producto Prent
descripcion	VARCHAR(256)			No	Descripción
descripcion_Corta	VARCHAR(128)				Descripción Corta
id_Empresa	DECIMAL(3)			Yes	Identificador Empresa
siglas	VARCHAR(8)			No	siglas
estatus	DECIMAL(1)	NULL			estatus

Figura 4.2.3.27 Tipo_Producto

Nobre de Tabla:	Evaluacion				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
numero_Producto	VARCHAR(18)	NOT NULL	Yes	Yes	numero Producto
fecha_Corte_Ciclico	DATE				fecha Corte Ciclico
id_Condicion	DECIMAL(2)				Identificador Condicion
id_Promocion	DECIMAL(6)				Identificador Promocion
id_Beneficio					Identificador Beneficio
CveTran					CveTran
GpoAfin					GpoAfin
Product	DECIMAL(9)				Product
id_Motivo_NA_Beneficio	DECIMAL(3)	NULL	No		Identificador Motivo NA Beneficio
estatus	DECIMAL(1)	NOT NULL		No	estatus

Figura 4.2.3.28 Evaluacion

Nobre de Tabla:	PreSuscripcion				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
id_Promocion	DECIMAL(6)		Yes	Yes	Identificador Promocion
id_PreSuscripcion	DECIMAL(7)			No	Identificador PreSuscripcion
nombre	VARCHAR(128)		No		nombre
apellido_Paterno					apellido Paterno
apellido_Materno					apellido Materno
direccion	VARCHAR(1024)				direccion
colonia	VARCHAR(128)				colonia
delegacion					delegacion
cp	VARCHAR(8)				cp
estatus	DECIMAL(1)	NULL			estatus

Figura 4.2.3.29 PreSuscripcion

Nobre de Tabla:	P_176_Concepto				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
CveTran	DECIMAL(6)	NOT NULL	Yes	Yes	CveTran
Sistema		NULL	No	No	Sistema
Descripcion Sistema	VARCHAR(128)				Descripción Sistema
Leyenda	DECIMAL(6)			Yes	Leyenda
Descripcion_Leyenda	VARCHAR(128)			No	Descripción Leyenda
Paramsi	DECIMAL(3)				Paramsi
Descripcion_Paramsi	VARCHAR(128)				Descripción Paramsi
N_Batch	DECIMAL(3)				N Batch
N_Linea					N Linea
N_Actua					N Actua
Fecha_Alta	DATE				Fecha Alta
Fecha_U_Mod					Fecha U Mod
Fecha_Vig					Fecha Vigente

Figura 4.2.3.30 P_176_Concepto

Nobre de Tabla:	D_58_Leyenda				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
CveLeye	DECIMAL(6)	NOT NULL	Yes		CveLeye
NumSist	CHAR(18)	NULL	No		NumSist
Descripcion_NumSist					Descripción NumSist
Descripcion_Larga					Descripción Larga
Descripcion_Corta					Descripción Corta
Descripcion_Pantalla					Descripción Pantalla
Abrev					Abreviación
Fecha_Alta					Fecha Alta
Fecha_U_Mod					Fecha U Mod

Figura 4.2.3.31 D_58_Leyenda

Nobre de Tabla:	Tipo_Concepto				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
CveTran	DECIMAL(6)	NOT NULL	Yes		CveTran
descripcion	VARCHAR(256)		No		Descripción
descripcion_Corta	VARCHAR(128)				Descripción Corta
siglas	VARCHAR(8)				siglas
estatus	DECIMAL(1)	NULL			estatus

Figura 4.2.3.32 Tipo_Concepto

Nobre de Tabla:	D_233_Producto				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
GpoAfin	DECIMAL(6)	NOT NULL	Yes		GpoAfin
Product	DECIMAL(9)				Product
Descripcion_Product	VARCHAR(128)	NULL	No		Descripción Product
Descripcion_Larga	VARCHAR(256)				Descripción Larga
Descripcion_Corta	VARCHAR(128)				Descripción Corta
Descripcion_Pantalla					Descripción Pantalla
Abrev	DECIMAL(6)				Abreviación
Fecha_Alta	DATE				Fecha Alta
Fecha_U_Mod					Fecha U Mod
Fecha_Baja					Fecha Baja

Figura 4.2.3.33 D_233_Producto

Nobre de Tabla:	D_38_Tipo_Tarjeta				
Nombre de Columna	**Tipo de dato de Columna**	**Opción de Columna Vacia**	**Columna es PK**	**Columna es FK**	**Descripción**
CveFlag	DECIMAL(3)	NOT NULL	Yes		CveFlag
Descripcion_Larga	VARCHAR(256)	NULL	No		Descripción Larga
Descripcion_Corta	VARCHAR(128)				Descripción Corta
Descripcion_Pantalla					Descripción Pantalla
Abrev	VARCHAR(10)				Abreviación
Fecha_Alta	DATE				Fecha Alta
Fecha_U_Mod					Fecha U Mod
Fecha_Baja					Fecha Baja

Figura 4.2.34 D_38_Tipo_Tarjeta

4.2.4 DIAGRAMA DE ENTIDAD RELACIÓN

El modelo entidad relación busca modelar los requerimientos en cuanto al almacenamiento de la información en el negocio. Para ser más preciso: trata sobre modelar los requerimientos de los datos para un negocio basado en la funcionalidad deseada para el sistema futuro. Para modelar el negocio se debe de tener un buen grado de conocimiento del detalle del funcionamiento del mismo. El generar un diagrama entidad relación es una técnica usada para describir el entendimiento de las necesidades de información que se tiene. Es una técnica bien definida que ayuda a generar diagramas fáciles de leer y de verificar.

Los objetivos del diagrama entidad relación son los siguientes:

- Describir exactamente la información que requiere el negocio.
- Facilitar la discusión acerca de las entidades a modelar, y llegar a un acuerdo.
- Ayuda a prevenir errores y evitar malos entendidos.
- Ayuda a generar un documento bien formado del sistema ideal.
- Es la base para el diseño físico de la base de datos.

El diagrama entidad relación contiene una serie de elementos que son:

- Entidad: Una entidad es cualquier “cosa” de interés para el negocio.
- Atributo: Es una característica que define a la entidad. La cual puede describir, cuantificar, calificar, clasificar, especificar, etc.
- Instancia: Un valor particular que se ajusta la definición de la entidad.
- Llave primaria: esta se encarga de identificar en forma única cada instancia de la entidad.
- Llave única: implementa reglas del negocio.

- Relación: Es una frase que especifica que es lo que hace una entidad con otra entidad.

En el apéndice se muestra el diagrama Entidad Relación completo para el sistema.

4.2.5 NORMALIZACIÓN

La normalización es un concepto aplicado a una base de datos relacional, es el proceso mediante el cual se transforman datos complejos a un conjunto de estructuras de datos más pequeñas, que además de ser más simples y estables, son fáciles de mantener; contribuyen a minimizar los problemas de lógica y consiste en aplicar una serie de reglas a las relaciones obtenidas tras el paso del modelo entidad relación al modelo relacional.

Las ventajas que se obtienen al aplicar la normalización son:

- Facilidad de uso. Los datos se agrupan en tablas que identifican visiblemente un objeto o una relación.
- Flexibilidad. La información necesaria puede obtenerse de las tablas relacionales o relaciones mediante las operaciones de álgebra relacional.
- Precisión. Las interrelaciones entre tablas mantienen información diferente relacionada con toda exactitud.
- Seguridad. Los controles de acceso para consultar o actualizar información son sencillos de implementar.
- Implementación. Las tablas se almacenan físicamente como archivos planos.
- Independencia. Los programas no están ligados a las estructuras, pudiendo incrementar la base de datos añadiendo nuevos atributos o nuevas tablas sin afectar los programas que las usan.
- Claridad. La representación de la información en tablas simples es clara y sencilla para el usuario.
- Gestión. Los lenguajes manipulan la información de manera sencilla al estar los datos basados en el álgebra y el cálculo relacional.
- Redundancia. La información no se duplica dentro de las estructuras.
- Rendimiento. Se trata sólo la información que se utilizará en cada aplicación.

En el modelo relacional se llama tabla a una relación, y tiene que cumplir con algunas restricciones:

- Cada columna debe tener su nombre único.
- No puede haber dos filas iguales. No se permiten los duplicados.
- Todos los datos en una columna deben ser del mismo tipo.

Una clave o llave primaria es una columna que identifica únicamente a esa fila. En una tabla puede haber más de una llave por lo que se puede escoger una para ser la llave primaria siendo las demás llaves foráneas. Una llave foránea es una columna dependiente en una tabla y a su vez llave primaria en otra tabla.[13] A la figura 4.2.5.1 que no está normalizada se aplicarán los conceptos de normalización.

USUARIO_ ID	NOMBRE	AP_ PAT	AP_ MAT	RFC	CALLE	COLONIA	CIUDAD	ESTADO
UABE01	RUTH	ABREU	ZAPATA	ABZR920619	BUGAMBILIAS 54	JARDINES DEL SOL	MEXICO	D.F.
1332055	LIZETH	DE LA CRUZ	CHAVEZ	DECL930615	PINO 125	BELLAVISTA	GOMEZ PALACIO	DURNANGO
0900076	EUGENIA	HERNANDEZ	SILVA	HESE800907	GUADALQUIVIR 1231	ESTRELLA	TORREON	COAHUILA
0645	FERNANDO	MARTINEZ	SANCHEZ	MASF850915	CALZ. AVILA CAMACHO 1503	NAVARRO	TORREON	COAHUILA
1044096	GERARDO	REYES	PIMIENTEL	REPG900612	ESCOBEDO 1916 OTE	CENTRO	COAHUILA	COAHUILA

[13] http://es.wikipedia.org/wiki/Clave_for%C3%A1nea

	PAIS	CP	TEL	CELBIP	SEXO	CATEGORIA	TIPO	RESERVA	MULTA	REG	TITULO	
	MEXICO	52789	5587-8784	55-2456-8758	F	U	AL	0	0.00			
	MEXICO	35050	714-4623		F	U	01	1	0.00	022236	DANZA MODERNA	
	MEXICO		21-78-30		F	U	01	1	15.00	120259	ASESINATO EN SENADO DE LA NACION	
	MEXICO		13-52-97	55-7375-5986	M	U	PR	0	0.00			
	MEXICO	27000	718-3125		M	U	01	2	0.00	160523 004926	CALCULO AVANZADO GEOMETRIA ANALITICA	

	AUTOR	CLASIFICACION	MAT	FECHA	DIAS
	DALLAL, ALBERTO	N 8217 D3	BK	1974	5
		120259	VM	1978	3
	FULKS, WASTON	QA 303 F954	BK	1970	5
	FULLER, GORDON	QA 551 F87	BK	1995	5

Figura 4.2.5.1 Control Bibliotecario

Formas normales

Existen varios niveles de normalización, para este proyecto se utilizaron la Primera Forma Normal (1FN), La Segunda Forma Normal (2FN) y la Tercer Forma Normal (3FN) debido a que éstas proveen suficiente nivel de normalización. Cada regla está basada en la que le antecede y para que una tabla este en un nuevo nivel o forma de normalización debe haber cumplido la regla anterior

Primera Forma Normal (1FN)

Una tabla está en Primera Forma Normal sólo si:

- Todos los atributos son indivisibles, mínimos.
- La tabla contiene una llave primaria.
- La tabla no contiene atributos nulos.
- Si no posee ciclos repetitivos.

Una columna no puede tener múltiples valores por lo que se eliminan valores repetidos dentro de una BD. La tabla 4.2.5.1 control bibliotecario quedará conformada en dos tablas que llamaremos Usuarios (ver tabla 4.2.5.2) y Circulación (ver tabla 4.2.5.3).

USUARIO_ ID	NOMBRE	AP_ PAT	AP_ MAT	RFC	CALLE	COLONIA	CIUDAD	
UABE01	RUTH	ABREU	ZAPATA	ABZR920619	BUGAMBILIAS 54	JARDINES DEL SOL	MEXICO	
1332055	LIZETH	DE LA CRUZ	CHAVEZ	DECL930615	PINO 125	BELLAVISTA	GOMEZ PALACIO	
0900076	EUGENIA	HERNANDEZ	SILVA	HESE800907	GUADALQUIVIR 1231	ESTRELLA	TORREON	
0645	FERNANDO	MARTINEZ	SANCHEZ	MASF850915	CALZ. AVILA CAMACHO 1503	NAVARRO	COAHUILA	
1044096	GERARDO	REYES	PIMIENTEL	REPG900612	ESCOBEDO 1916 OTE	CENTRO	COAHUILA	

ESTADO	PAIS	CP	TEL	CELBIP	SEXO	CATEGORÍA	TIPO
D.F.	**MEXICO**	52789	5587-8784	55-2456-8758	F	U	AL
DURANGO	**MÉXICO**	35050	714-4623		F	U	01
COAHUILA	**MEXICO**		21-78-30		F	U	01
COAHUILA	**MEXICO**		13-52-97	55-7375-5986	M	U	PR
COAHUILA	**MEXICO**	27000	718-3125		M	U	01

Tabla 4.2.5.2 Usuarios

USUARIO_ ID	MULTA	RESERVA	DIAS	REG	TITULO	AUTOR	CLASIFICACION	MAT	FECHA
UABE01	0.00	0							
1332055	0.00	1	5	022236	DANZA MODERNA	DALLAL, ALBERTO	N 8217 D3	BK	1974
0900076	15.00	1	3	120259	ASESINATO EN SENADO DE LA NACION		120259	VM	1978
0645	0.00	0							
1044096	0.00	2	5 5	160523 004926	CALCULO AVANZADO GEOMETRIA ANALITICA	FULKS, WASTON FULLER, GORDON	QA 303 F954 QA 551 F87	BK BK	1970 1995

Tabla 4.2.5.3 Circulación

Segunda Forma Normal (2FN)

Una relación está en esta forma si está en 1FN y si los atributos que no forman parte de ninguna llave dependen de forma completa de la llave principal. En esta regla todas las dependencias parciales deben eliminarse y separarse dentro de sus propias tablas. Una

dependencia parcial es un término que describe aquellos datos que no dependen de la llave primaria para identificarlos.

La tabla 4.2.5.3 Circulación no se encuentra en 2FN, ya que las columnas referentes a los datos del material son independientes de reserva, por lo tanto eliminaremos esas columnas y crearemos la tabla de Préstamos (ver tabla 4.2.5.4).

USUARIO_ ID	RESERVA	DIAS	MULTA
UABE01	0		0.00
1332055	1	5	0.00
0900076	1	3	15.00
0645	0		0.00
1044096	2	5 5	0.00

Tabla 4.2.5.4 Préstamos

Tercera Forma Normal (3FN)

La tabla se encuentra en esta forma si está 2FN y todas las columnas que no son llave son funcionalmente dependientes por completo de la llave primaria y no hay dependencias transitivas. Una dependencia transitiva es aquella en la cual existen columnas que no son llaves que dependen de otras columnas que tampoco son llaves.

Las tablas reservación (ver tabla 4.2.5.4) y préstamos (ver tabla 4.2.5.5) se encuentra en 3FN, sin embargo la tabla clientes (ver tabla 4.2.5.2) no lo está, ya que NOMBRE y RFC son independiente de USUARIO_ID. Para normalizarla moveremos las columnas a una nueva tabla llamada directorio (ver tabla 4.2.5.5).

USUARIO_ ID	NOMBRE	RFC
UABE01	RUTH	ABZR920619

1332055	LIZETH	DECL930615
0900076	EUGENIA	HESE800907
0645	FERNANDO	MASF850915
1044096	GERARDO	REPG900612

Tabla 4.2.5.5 Directorio

4.3 DISEÑO Y CONSTRUCCIÓN DEL BACK – END

Back-End hace referencia a la parte de la aplicación que determina el almacenamiento de la información y la organización de las tablas, el objetivo es crear la estructura del sistema modular para los procesos de captura de datos, búsqueda de información y actualización.

La versión que se utilizará de Firebird será la 1.5 por ser la más estable, además de que es multiplataforma. Firebird tiene dos arquitecturas, Classic Server y Superserver, en la siguiente tabla (Tabla 4.3.1) se mencionan las diferencias entre las dos.

Classic Server	Superserver
Totalmente estable en Linux pero aún "experimental" en Windows	Totalmente estable tanto en Linux como en Windows
Permite entrada / salida directa, rápida a archivos de bases de datos para conexiones locales (sólo Linux).	Las conexiones locales deben hacerse con la forma de acceso remoto, conectando a *localhost.* En Windows se pueden hacer conexiones locales, pero no son tan veloces como las de versión "Classic" en Linux además de ser menos seguras.
Windows: Services Manager (Administrador de servicios) implementados parcialmente, tareas de soporte como backup/restore, database shutdown, a través de la red. Otras tareas administrativas tienen que ser realizadas localmente usando las herramientas de	Administrador de Servicios completo (Windows y Linux) que permite realizar tareas de administración (backup/restore, database shutdown, manejo de usuarios, estadísticas, etc.). El administrador de servicios a través de la red y realizar tareas remotamente.

cliente de Firebird. Linux: Administrador de Servicios completo.	
Soporte para SMP (Multi-procesador). Mejor rendimiento en caso de un pequeño número de conexiones simultáneas.	No hay soporte para SMP. En máquinas multiprocesador con Windows, el rendimiento puede incluso ser dramáticamente menor cuando el SO cambia el proceso entre los procesadores.

Tabla 4.3.1 Classic Server vs Supererver

Con lo anterior se puede concluir que ninguna de las dos arquitecturas es mejor en todos los aspectos, sin embargo el fabricante recomienda lo siguiente: si la instalación es sobre Windows se deberá instalar “Superserver”, si la instalación es sobre Linux, entonces podrían ser cualquiera de las dos, ya que en la mayoría de los casos no se notará la diferencia en rendimiento. Sin embargo en cualquier momento el usuario puede cambiar de una arquitectura a otra, las aplicaciones y bases de datos seguirán funcionando sin ningún problema, salvo que las aplicaciones manden llamar funciones no soportadas o no completadas del Administrador de Servicios en Classic.
(Figura 4.3.2)

Plataforma	Componente	Nombre de Archivo	Ubicación por defecto
Windows 32 - bit y 64 – bit. (Windows 95, 98, ME, NT, 2000, XP,…	Directorio de instalación. Se refiere en la explicación como <InstallDir>		C:\Archivos de programa\Firebird\Firebord_1_5
	Servidor Firebird	fbserver.exe (SS SuperSever) o	<InstallDir>\bin

		fb_inet_server.exe (CS Classic Server)	
	Herramientas de línea de comandos	gbak.exe, gfix.exe, gstat.exe, etc.	<InstallDir>\bin
	Base de datos de ejemplo	Employee.fbd	<InstallDir>\bin
	Librerías de funciones definidas por el usuario (UDF)	ib_idf.dll & fbud.dll	<InstallDir>\UDF
	Cliente Firebird	Fbclient.dll	<InstallDir>\bin
Linux y posiblemente otras distribuciones de UNIX	Directorio de instalación. <InstallDir>		/opt/firebird
	Servidor Firebird	fbserver (SS Superserver) o fb_inet_server (CS Classic Server)	<InstallDir>/bin
	Herramientas de línea de comandos	gbak, gfix, gstat, etc.	<InstallDir>/bin
	Base de datos de ejemplo	Employee.fdb	<InstallDir>/examples
	Librerías UDF	Ib_udf.so, fbudf.so	<InstallDir>/UDF
	Cliente Firebird	Libfbclient.so.1.5.n (binario);	/usr/lib

		libfbclient.so.1, libfbclient.so (enlace simbólico)	

Figura 4.3.2 Instalación en Windows y Linux

Nota. Las rutas típicas para el directorio de Windows dependen de la versión de Windows, algunas ubicaciones típicas son:

- Win 95/98ME; C:\Windows\System.
- Win NT/2000; C:\WINNT\System32.
- Win XP; C\Windows\System32.

Instalación en Windows Firebird (Super Server)

Se instalará la versión de Firebird Super Server en Windows por recomendación misma del fabricante. (Figura 4.3.3)

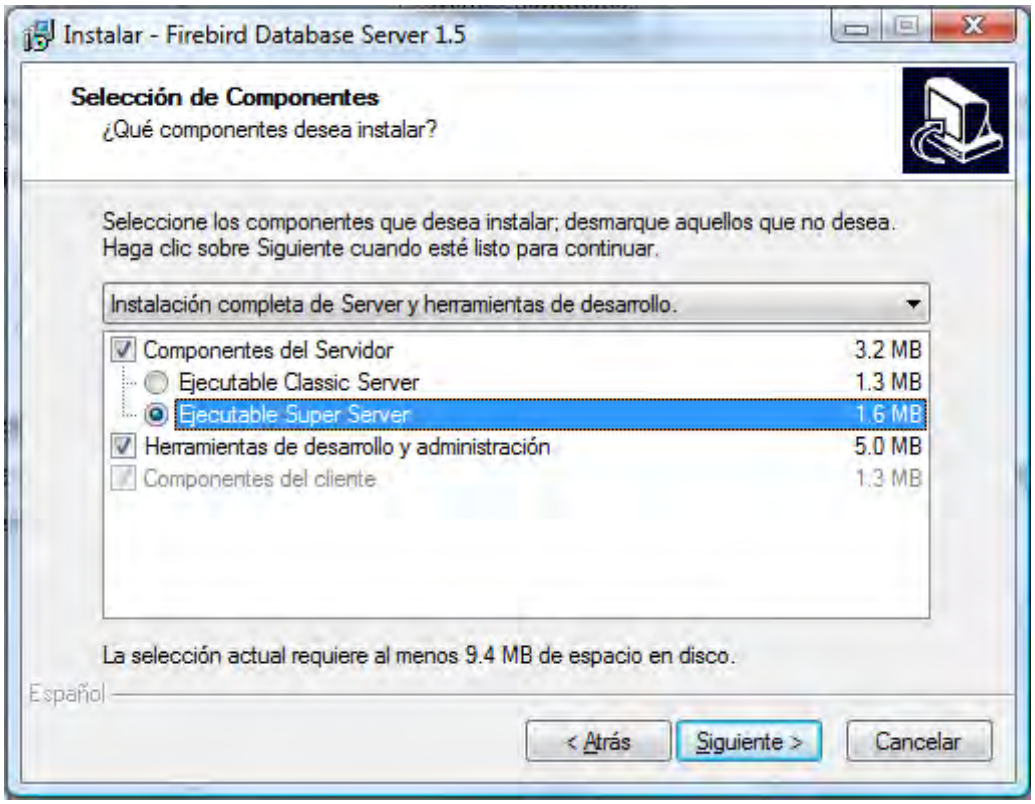


Figura 4.3.3 Instalación de Firebird en Windows

Firebird debe de ejecutarse como un servicio por lo tanto se verifica que el servicio se encuentre activo, para ello debemos de verificar los servicios de Windows, y corroborar que Firebird Guardian se encuentra iniciado. (Figura 4.3.4).

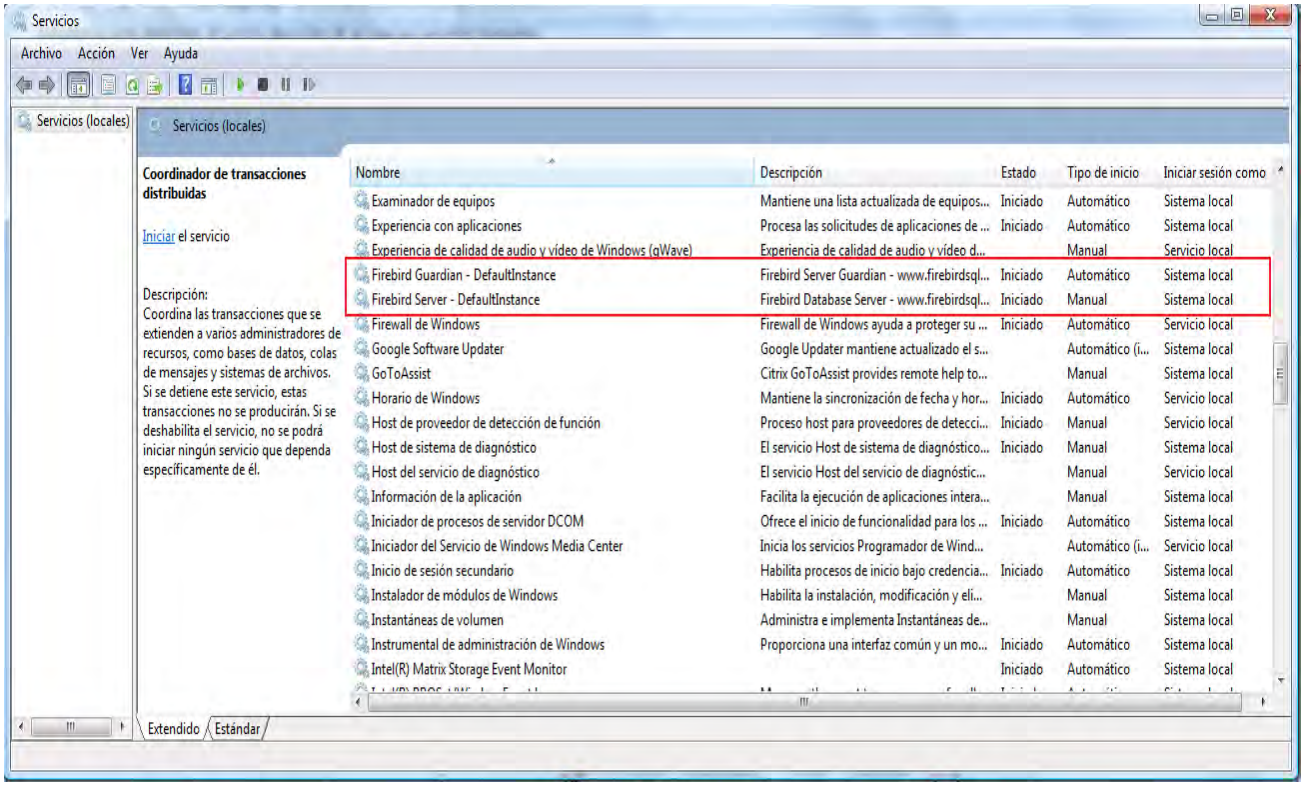


Figura 4.3.4 Servicios de Windows

Conectando a la base de datos de ejemplo

En la instalación de Firebird se creó un directorio de examples y una base de datos de ejemplo llamada employee.fdb, para realizar la conexión a dicha base se debe ingresar al directorio donde se tienen los comandos para la administración de Firebird.

- Para Linux: servidor:/ruta_al_archivo/archivo_de_la_base_de_datos.
- Para Windows. letra_de_disco:\ruta\archivo_de_base_de_datos.

En la figura 4.3.5 se muestran los comandos para la conexión a la base de datos de prueba.

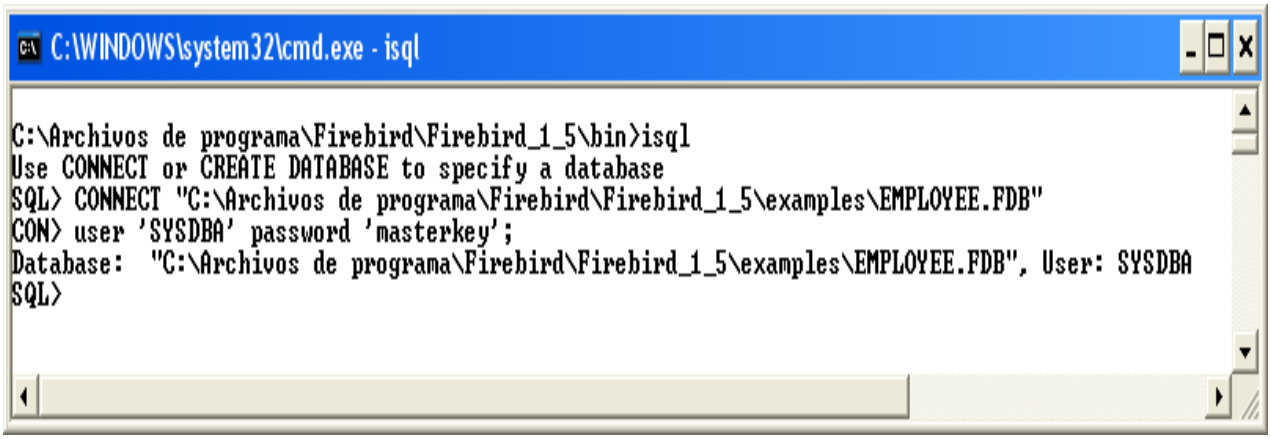


Figura 4.3.5 Conexión a la base de datos de prueba de Firebird

Creación de una base de datos en Firebird

Para crear una base de datos en forma interactiva usando la interfaz de comandos de isql, se debe de trabajar en el servidor. Debemos posicionar la consola de comandos en el subdirectorio bin y arracar el servicio isql, suponemos que creamos la base de datos biblioteca y se guardará en la raíz del disco. La base de datos biblioteca es creada inmediatamente después el cursor de SQL volverá aparecer, eso quiere decir que se encuentra conectado a la nueva base de datos, para corroborar lo anterior se ejecuta un select sobre el objeto RBD$RELATIONS, el objeto contiene los metadatos de las tablas. Figura 4.3.6.

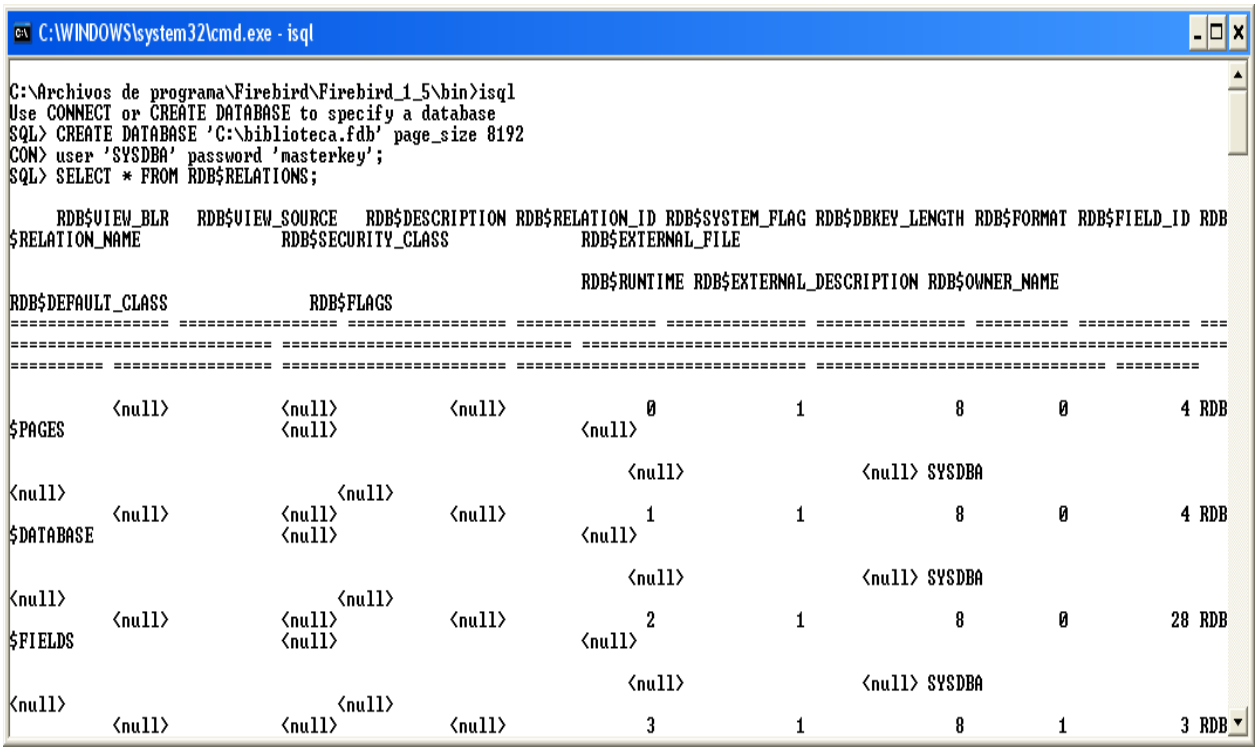


Figura 4.3.6 Creación de una base de datos en Firebird

Una vez que se tiene la base de datos creada y nos encontramos conectados a dicha base, creamos una tabla llamada Usuario, con los campos de idUsuario, Nombre y Apellido, insertamos un registro y realizamos el select de dicho registro para verificar que la información fue agregada correctamente. (Figura 4.3.7).

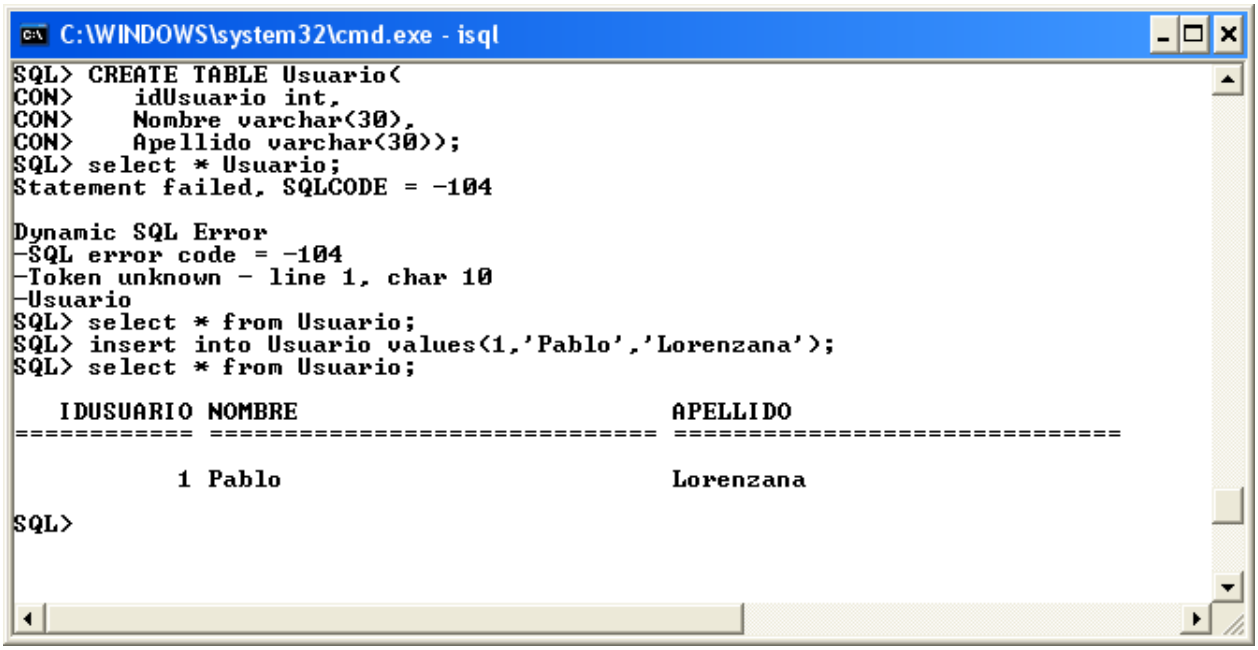


Figura 4.3.7 Creación de tabla e inserción de un registro

El manejador de Firebird se maneja totalmente por medio de SQL, sin embargo SQL se encuentra dentro de los estándares de ANSI (American National Standards Institute) existen algunas diferencias entre cada manejador de SQL. Sin embargo los comandos como SELECT, UPDATE, DELETE, INSERT, WHERE son similares para todos los manejadores.

Herramientas gráficas para la administración de la Base de Datos

Es difícil la administración por medio de comandos de sql, aunque es posible es poco práctica, por lo tanto existen herramientas gráficas que facilitan la administración de la base de datos. Específicamente para Firebird existe una herramienta que no pertenece al proyecto Firebird sino a otro grupo de software llamado Hk-Software y el producto que se usará será IB Expert en su versión de prueba.

La versión de prueba de IB Expert no solicita licencia, sin embargo, la versión de "cliente" en la cual se debe pagar, ofrece más herramientas para una óptima administración. Para nuestros fines con la versión de prueba es suficiente.

IB Expert administración gráfica de Firebird

Una vez que se tiene instalada la versión de IB Expert, se debe de registrar una base de datos, para el caso de estudio se registrará la base BIBLIOTECA.FDB. Figura 4.3.8.

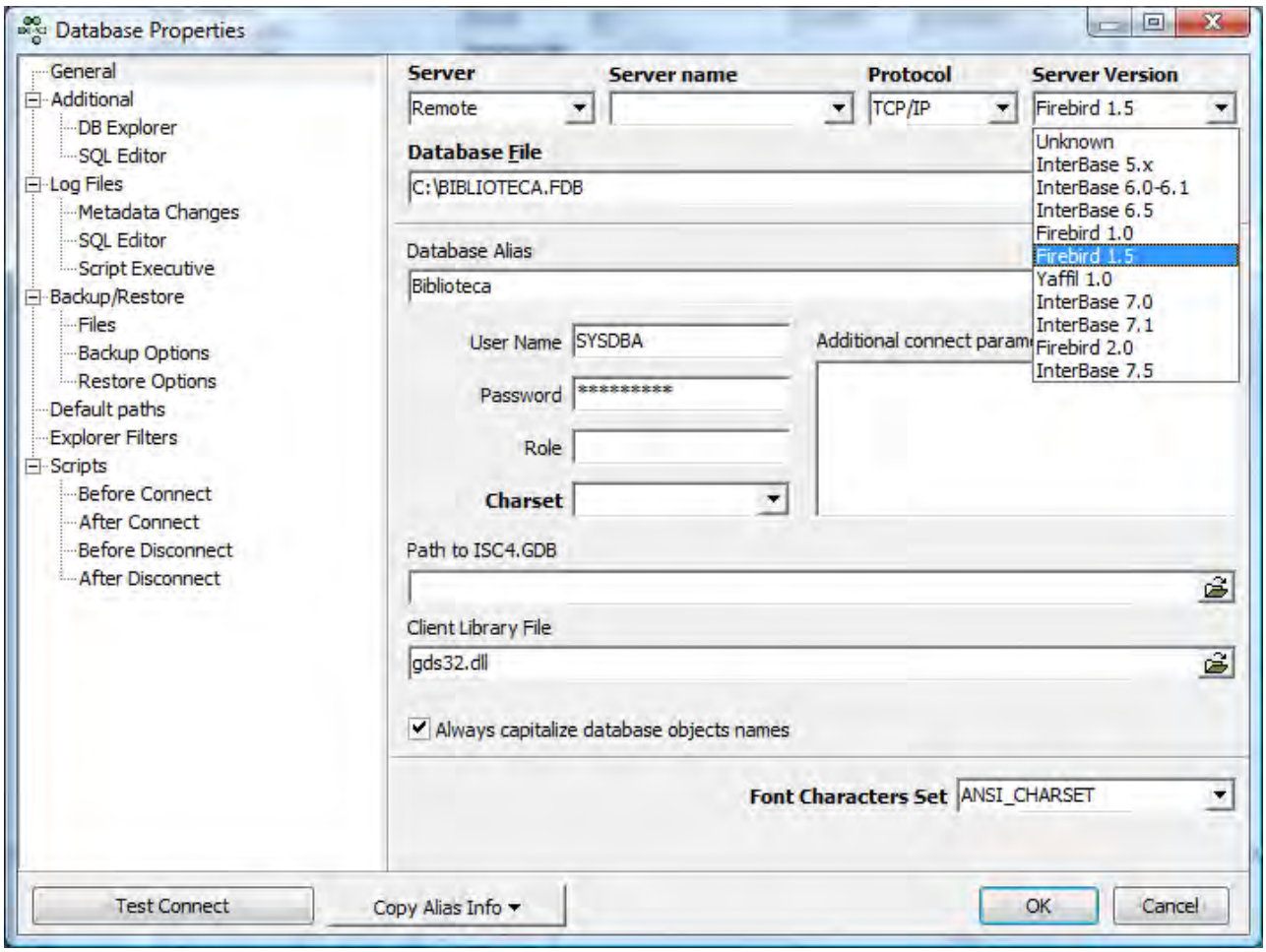


Figura 4.3.8 Registro de la base de datos en IB Expert para Firebird

Una vez que la base se importa en IB Expert, nos mostrará las tablas con las que cuenta la base, triggers, vistas y procedimientos almacenados. Figura 4.3.9.

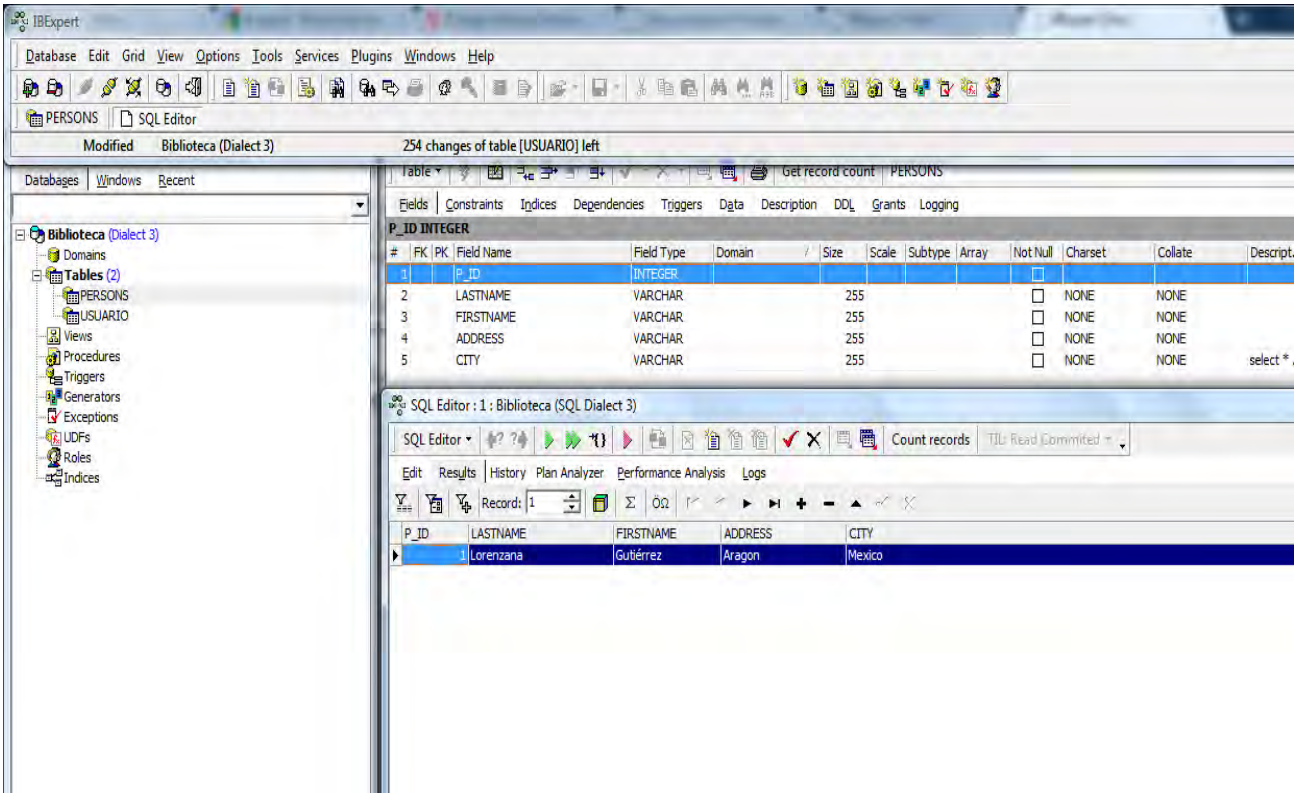


Figura 4.3.9 Vista de IB Expert con la base de datos importada

Para la creación de nuevas tablas desde la aplicación gráfica se debe dar clic sobre la opción “Tables” del menú izquierdo, en él aparecerá la opción “New Table” inmediatamente después que se haya presionado aparece una ventana donde se deben llenar los campos de la nueva tabla. Figura 4.3.10.

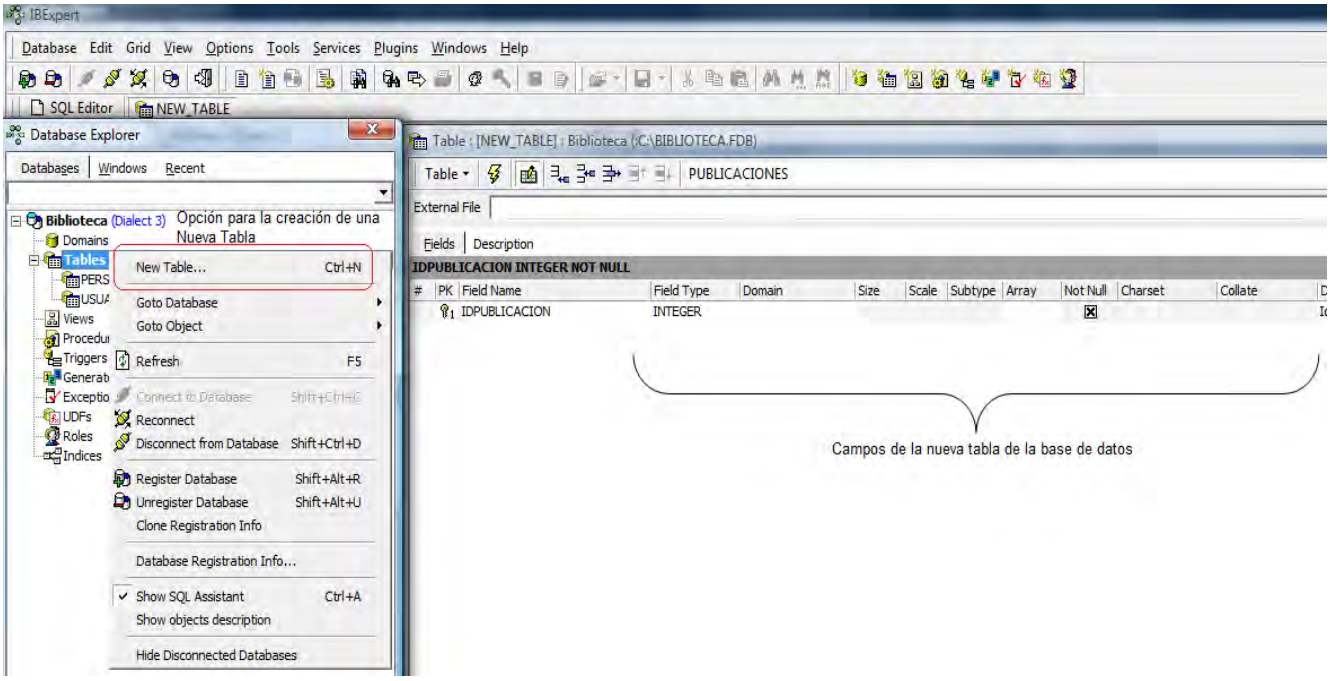


Figura 4.3.10 Creación de una nueva tabla en IB Expert

Para realizar una consulta sobre alguna tabla, solamente debemos posicionarnos en la ventana de edición y presionar sobre el botón “Data”, en la parte inferior se desplegarán los datos. Figura 4.3.11.

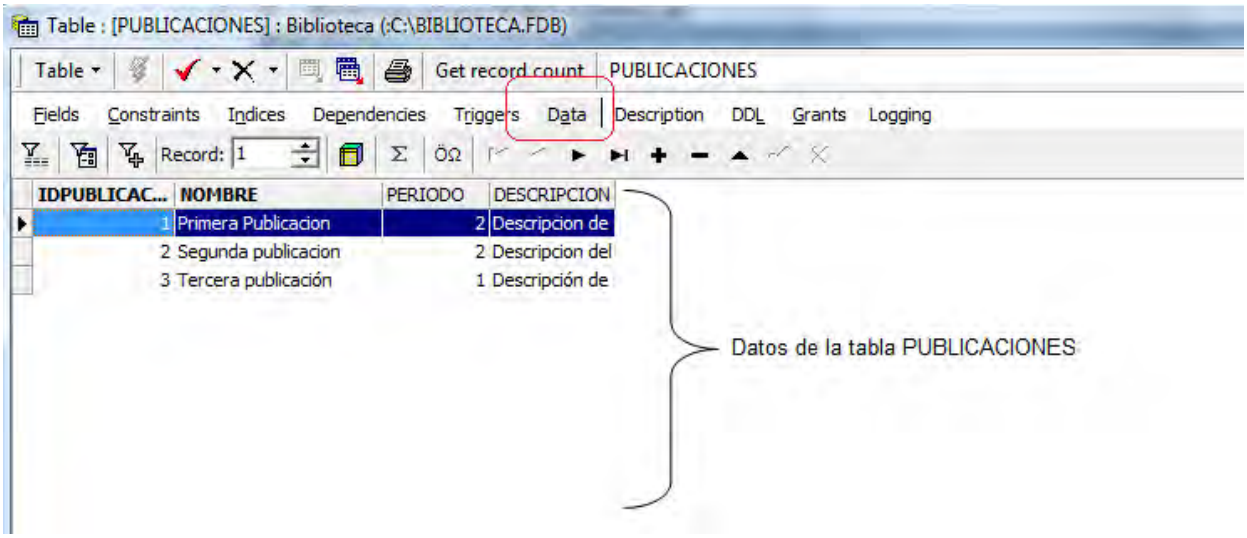


Figura 4.3.11 Datos almacenados en la tabla PUBLICACIONES

Si nos interesa crear sentencias de SQL más complejas, entonces se dispone de un editor de SQL, en el menú principal. Figura 4.3.12.

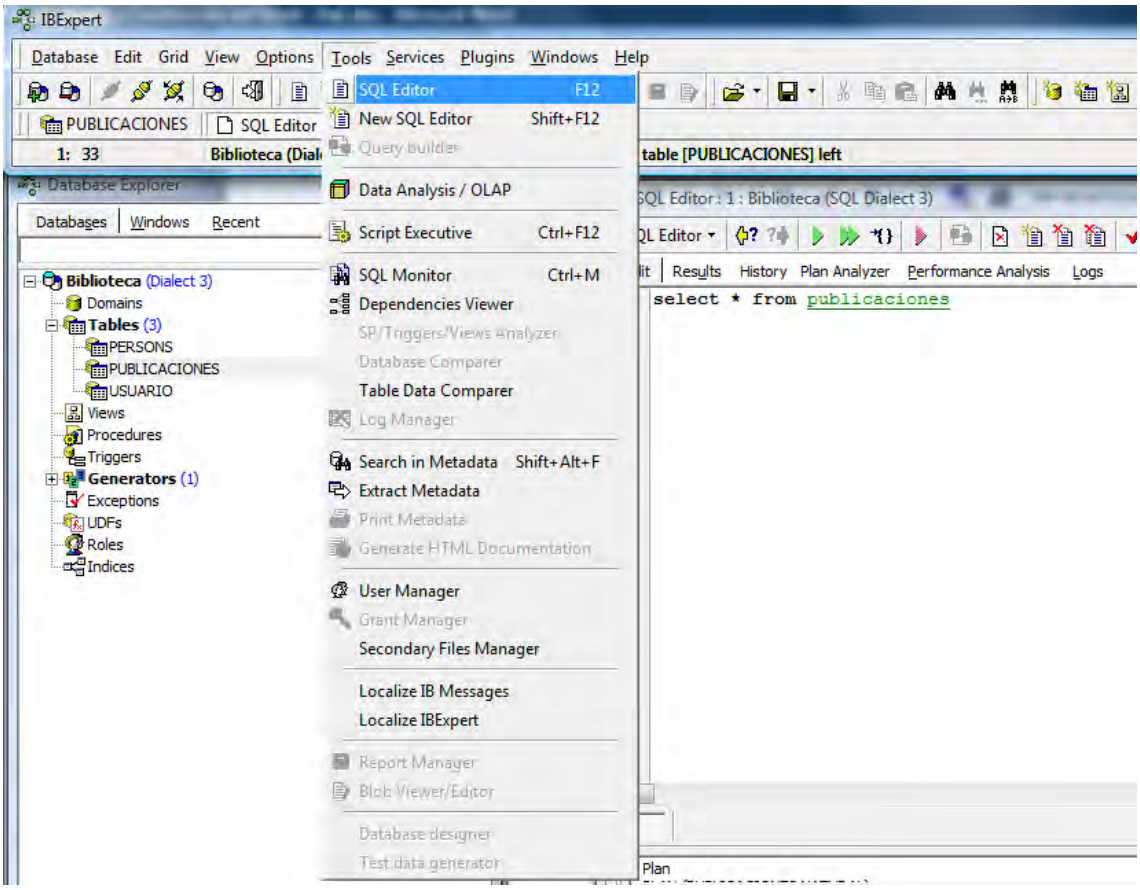


Figura 4.3.12 SQL Editor de IB Expert

Una vez que se dispara la ventana de SQL Editor, nos posicionamos en la ventana Edit y podemos escribir las sentencias de SQL tan complejas como se desee, después seleccionar "execute" o F9, para ejecutar la sentencia y el resultado se arroja en la pestaña de "Results". Figura 4.3.13.

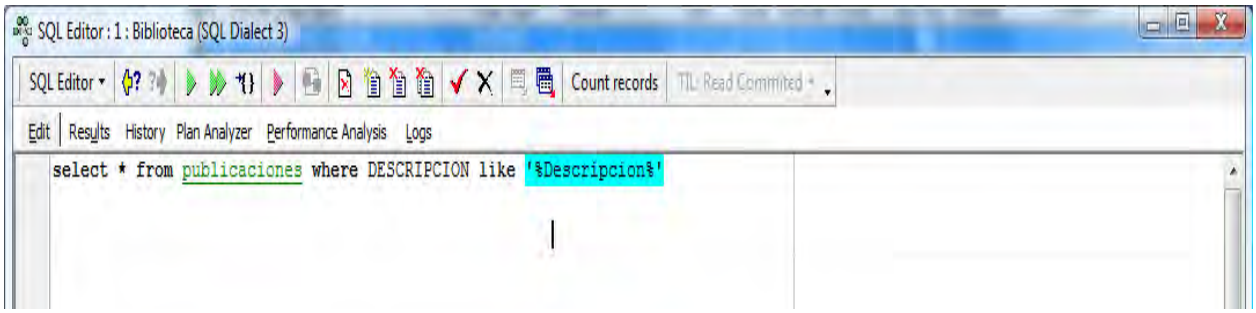


Figura 4.3.13 Sentencia escrita y ejecutada en SQL Editor

En la figura (Figura 4.3.14) se muestra el árbol de tablas en el menú izquierdo, en la ventana superior derecha se muestra la estructura de la tabla “Adeudo” y en la ventana inferior izquierda se muestran los datos de la tabla “Copias”.

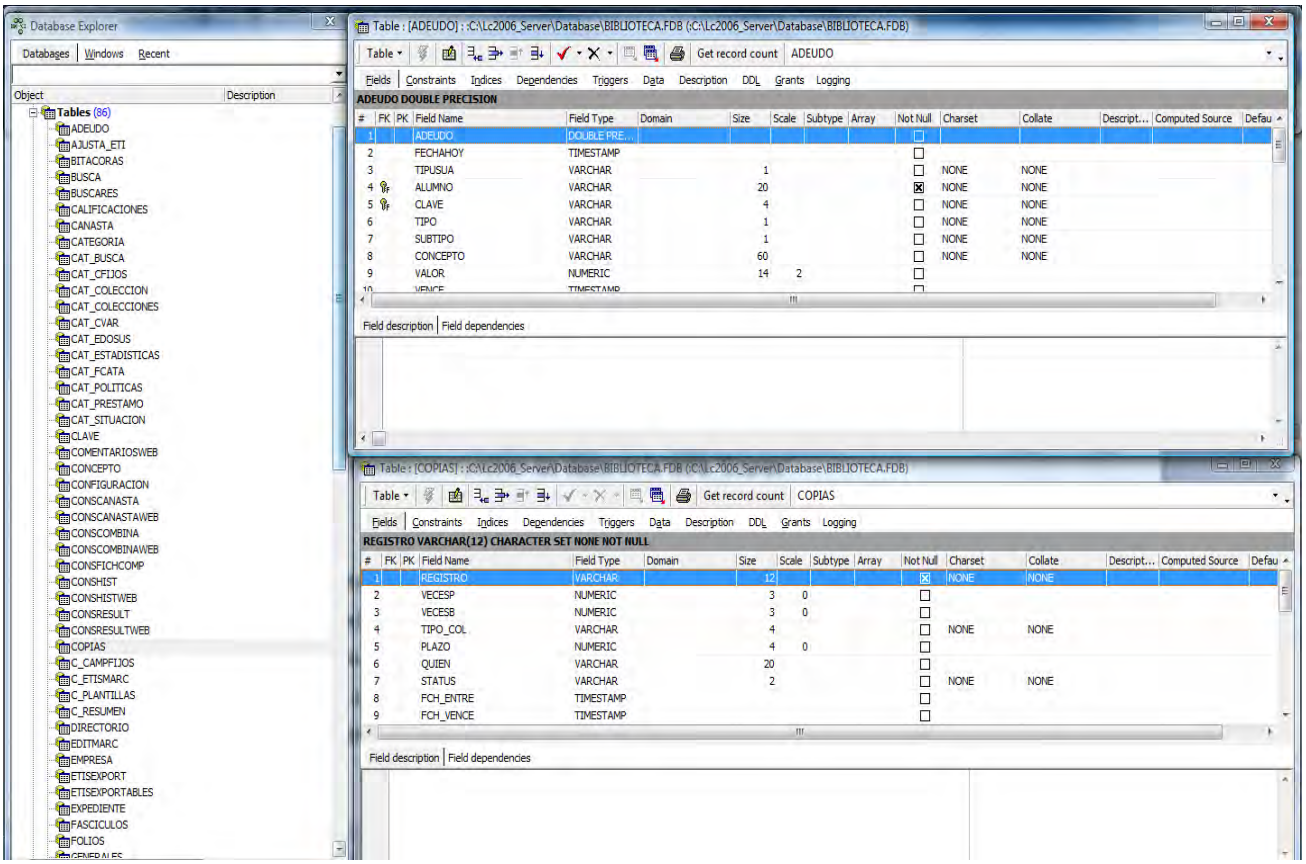


Figura 4.3.14 Árbol de tablas del Sistema de Control Bibliotecario

Igualmente con la herramienta se pueden crear vistas, triggers y procedimientos almacenados.

Para la creación de una nueva vista, se seleccionará la opción “Views” del menú izquierdo e inmediatamente después seleccionar la opción “New View” del menú desplegable. Figura 4.3.15.

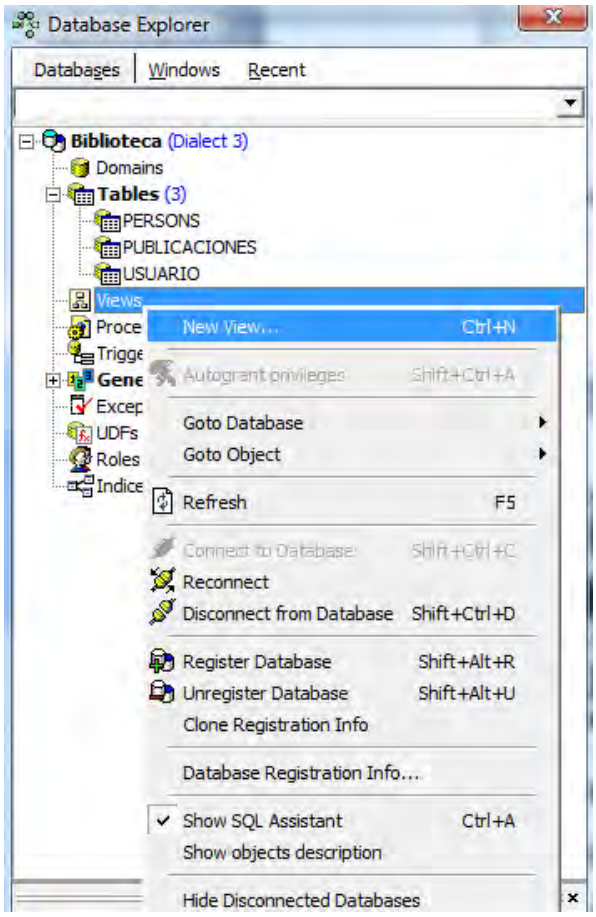


Figura 4.3.15 Creación de una vista.

Se disparará la ventana “View” que nos dará las opciones para crear la vista, para nuestro objetivo se creará una vista sencilla donde nos muestre el contenido de dos tablas.

Una vez realizada la vista, el campo de “View” del menú izquierdo aparecerá que existe una vista en la base de datos. Figura 4.3.16.

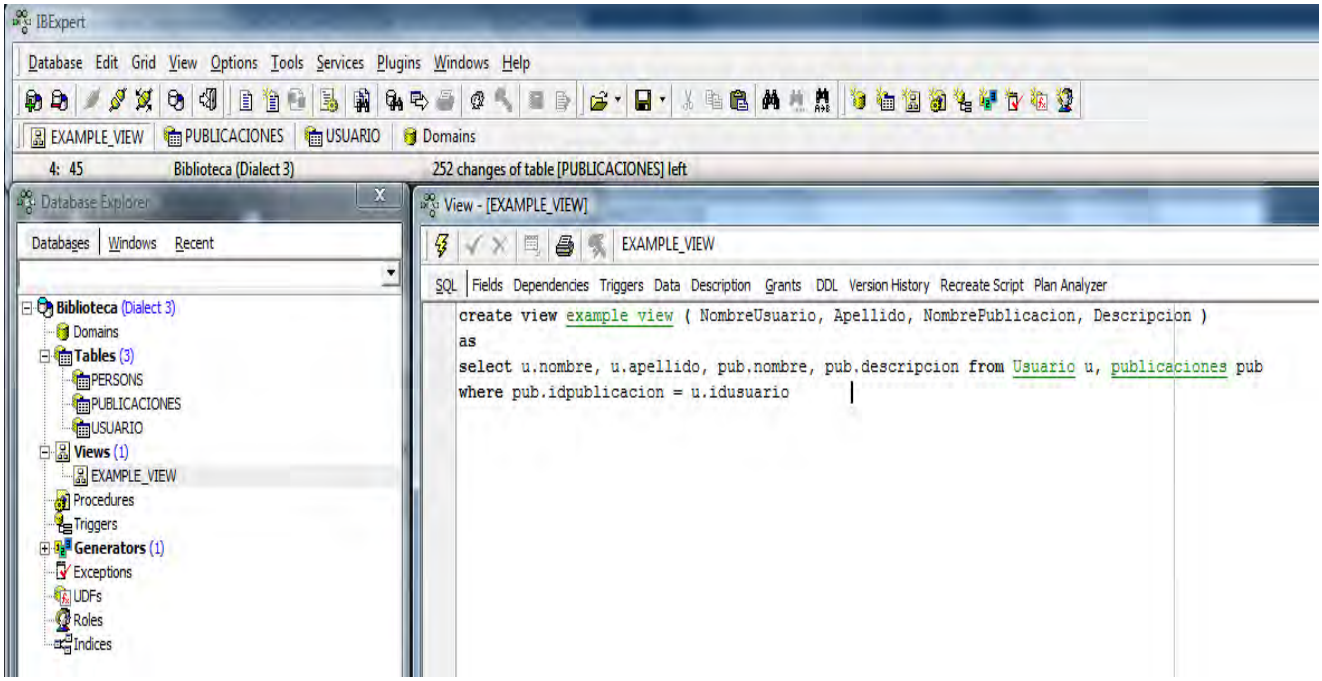


Figura 4.3.16 Editor de Vistas

Como en el caso de las tablas, si deseamos ver la información que arroja la vista presionamos sobre el botón “Data” de las pestañas superiores de la ventana “View”. Figura 4.3.17.

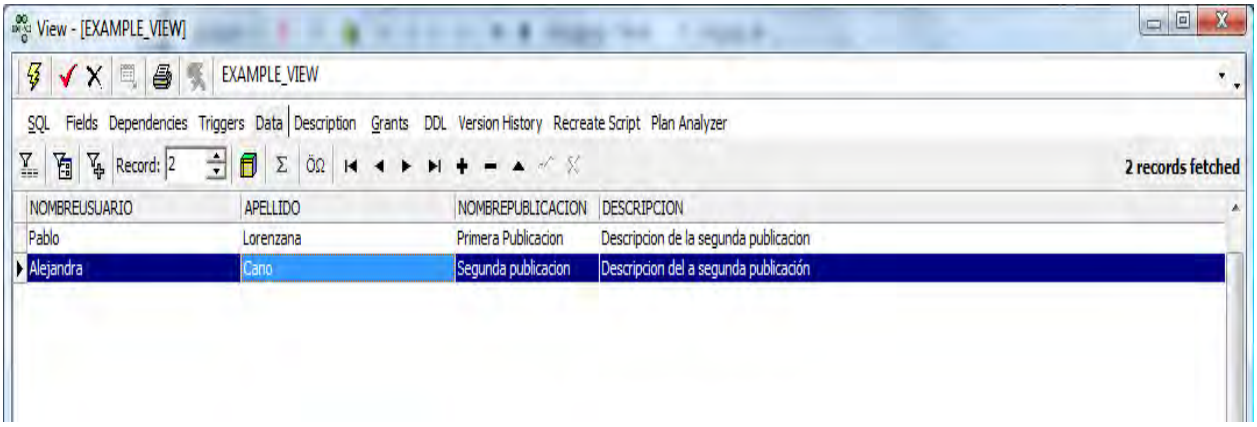


Figura 4.3.17 Datos que muestra la vista

El sistema de Bibliotecas cuenta con cinco vistas, en la figura 4.3.18 se muestra el árbol de vistas, en la ventana izquierda, la ventana superior derecha se muestra la definición de la vista y en la ventana inferior izquierda se muestran los datos que arrojan otra vista.

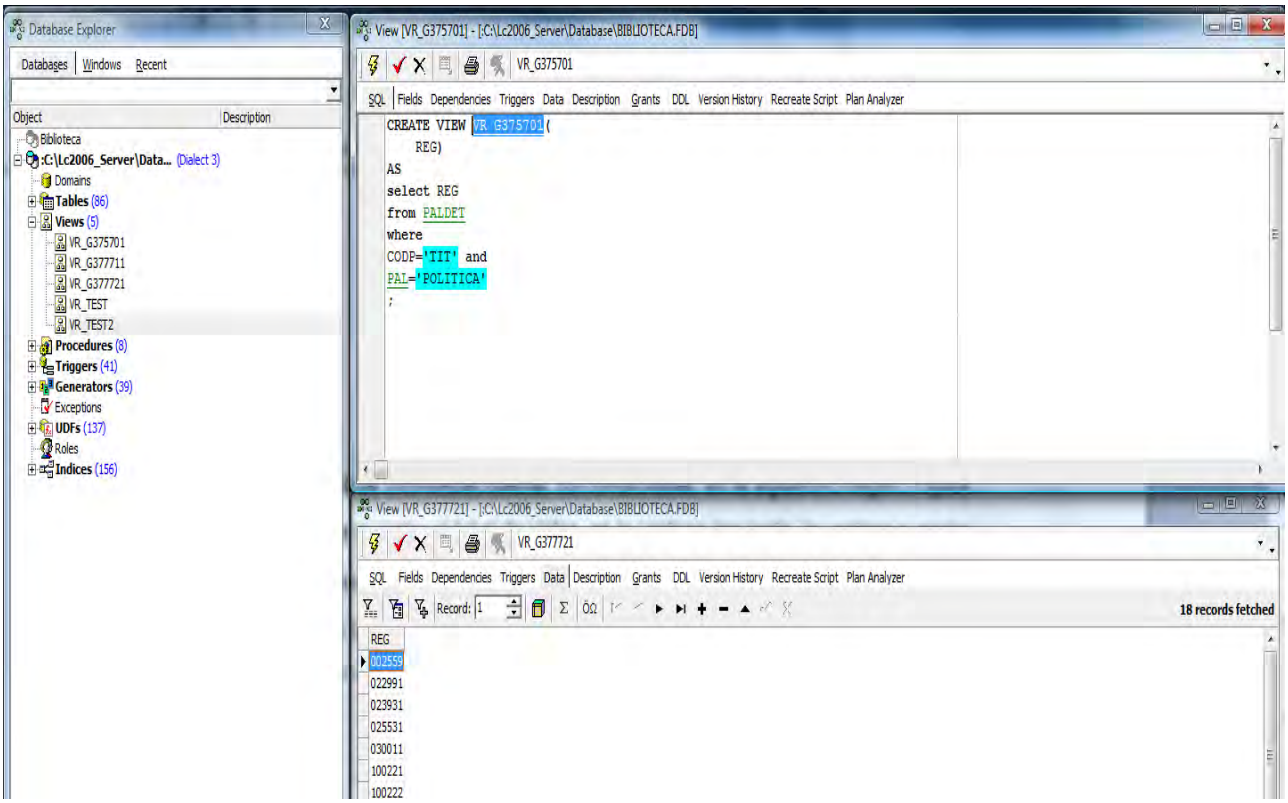


Figura 4.3.18 Árbol de vistas del Sistema de Control Bibliotecario

Para la creación de Triggers se realizan de forma similar que las vistas, los triggers son rutinas que son conectadas a una tabla o vista, se ejecuta automáticamente cuando un registro es insertado, actualizado o borrado. Los triggers tienen las siguientes ventajas:

- Ejecución automática, dependiendo de las condiciones con las que fue creado.
- Reduce el mantenimiento de las aplicaciones.
- Notificaciones automáticas a las aplicaciones pueden se llamadas por medio de eventos.

Para la creación de Triggers en IB Expert en el menú izquierdo se selecciona la opción de “Triggers” y se hace clic derecho para que se muestre el menú desplegable, una vez que se muestra el menú hacer clic sobre la opción “New Trigger…”. Figura 4.3.19.

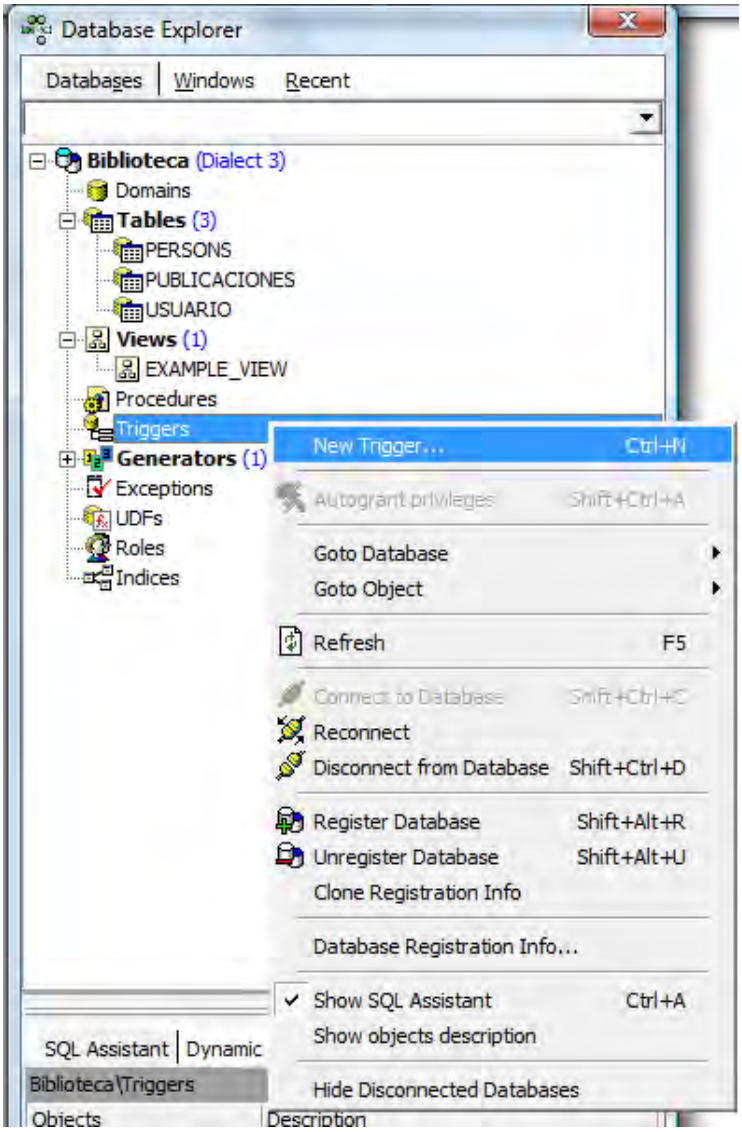


Figura 4.3.19 Creación de un Trigger

Igualmente aparecerá una ventana con titulo “Trigger” y un editor con las diferentes opciones para crear un trigger, supongamos que deseamos actualizar o insertar en la

tabla usuarios una vez que se haya actualizado o insertado datos en la tabla persons, el script quedaría como en la siguiente figura. Figura 4.3.20.

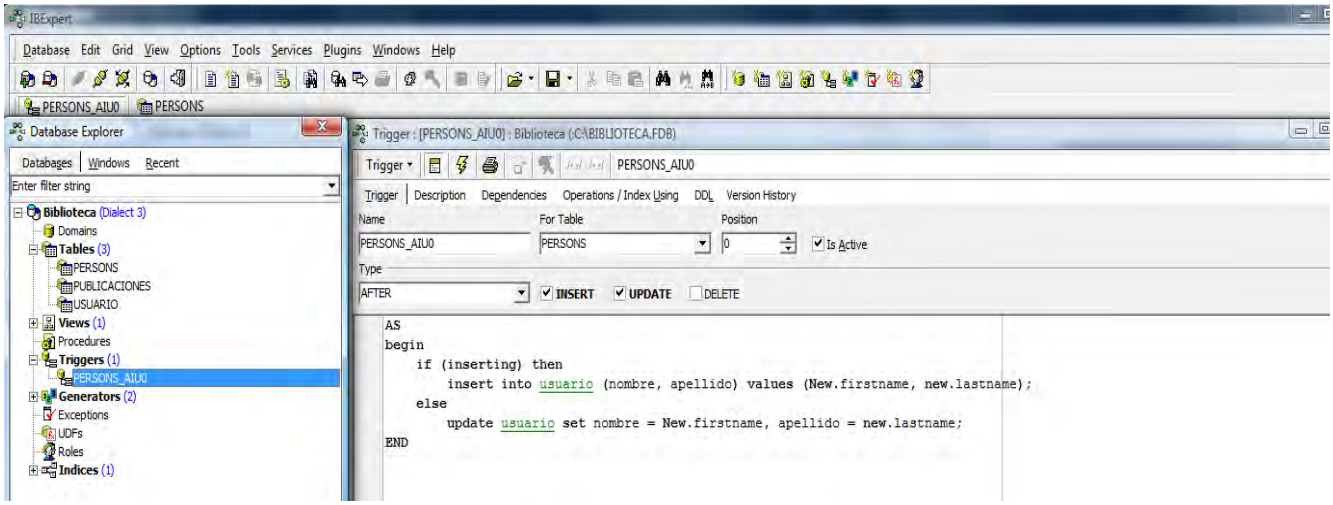


Figura 4.3.20 Script de la creación del Trigger

El sistema cuenta con 41 triggers que se muestran en la figura 4.3.21, el árbol se encuentra en la ventana izquierda, en la parte derecha se pueden observar la definición de dos vistas.

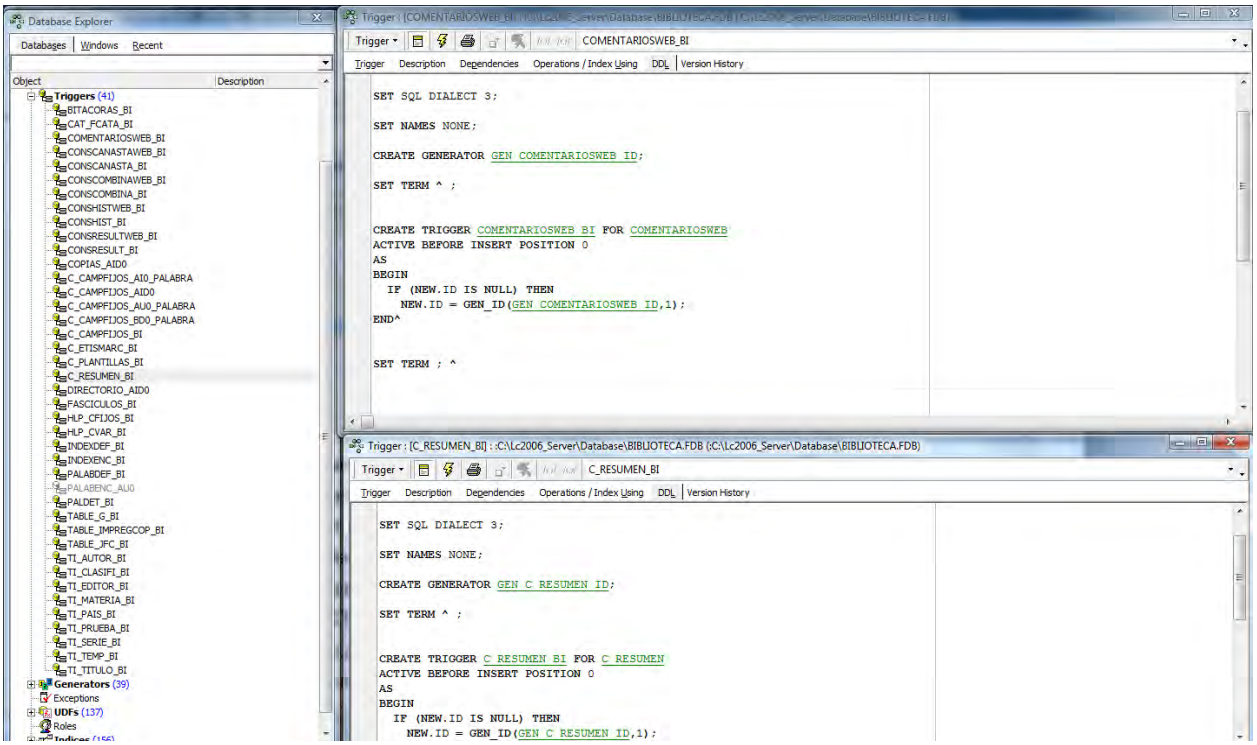


Figura 4.3.21 Árbol de Triggers del Sistema de Control Bibliotecario

Los Stored Procedures son rutinas que corren del lado del servidor y pueden ser llamados por los clientes, los stored son pre – compilados, por lo tanto solamente se invocan y son ejecutados. Estos procedimientos pueden tener parámetros de entrada y salida, dentro de ellos pueden ejecutarse sentencias como SELECT, UPDATE o cualquier instrucción de ANSI SQL. Los stored procedures tienen las siguientes ventajas.

- Las aplicaciones cliente son pequeñas y menos complicadas para ser implementadas y para ser mantenidas.
- Fácil mantenimiento. Las aplicaciones clientes no necesitan ser recompilados y redistribuidos, sino el cambio se realiza directamente desde la base de datos y los cambios son inmediatos en todos los clientes.
- Incrementa el rendimiento porque reduce el tráfico de información en la red.

Los procedimientos almacenados (Stored Procedures) pueden actuar como una acción y no regresar información, o pueden regresar tablas o vistas. Para ejecutar el procedimiento, el usuario que lo manda llamar deberá tener permisos de ejecución sobre el procedimiento específico.

Para la creación de un procedimiento almacenado en IB Expert en el menú izquierdo se selecciona la opción “Procedures” y se hace clic derecho para que sea mostrado el menú desplegable y hacer clic sobre la opción “New Procedure…”. Figura 4.3.22.

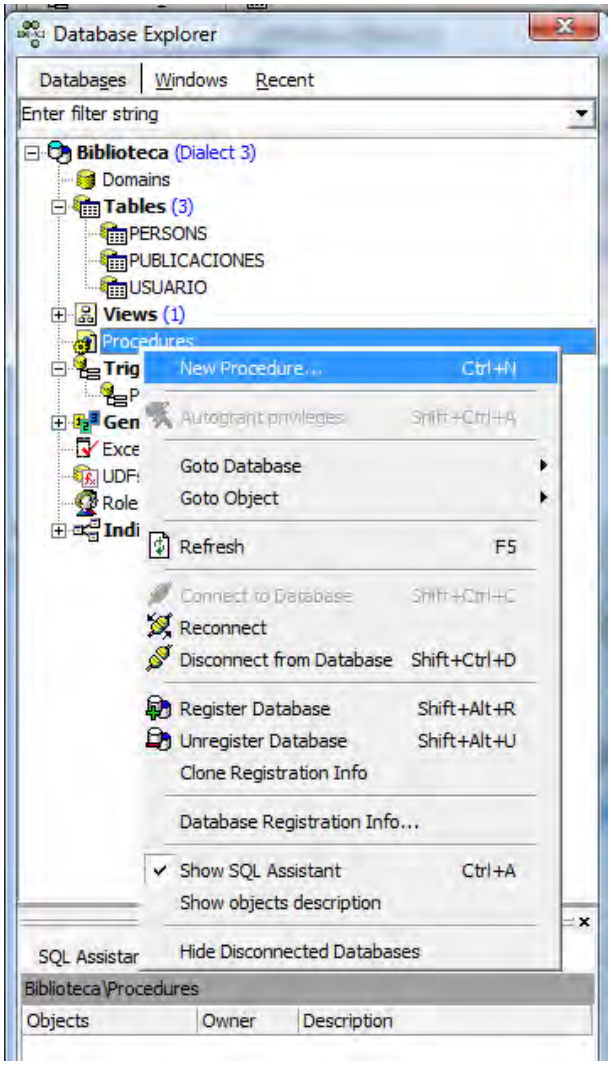


Figura 4.3.22 Creación de un Stored Procedure

Se abre la ventana de “Procedure” y se deben ingresar los parámetros de entrada, los parámetros de salida y las variables que serán usadas dentro del procedimiento. Figura 4.3.23.

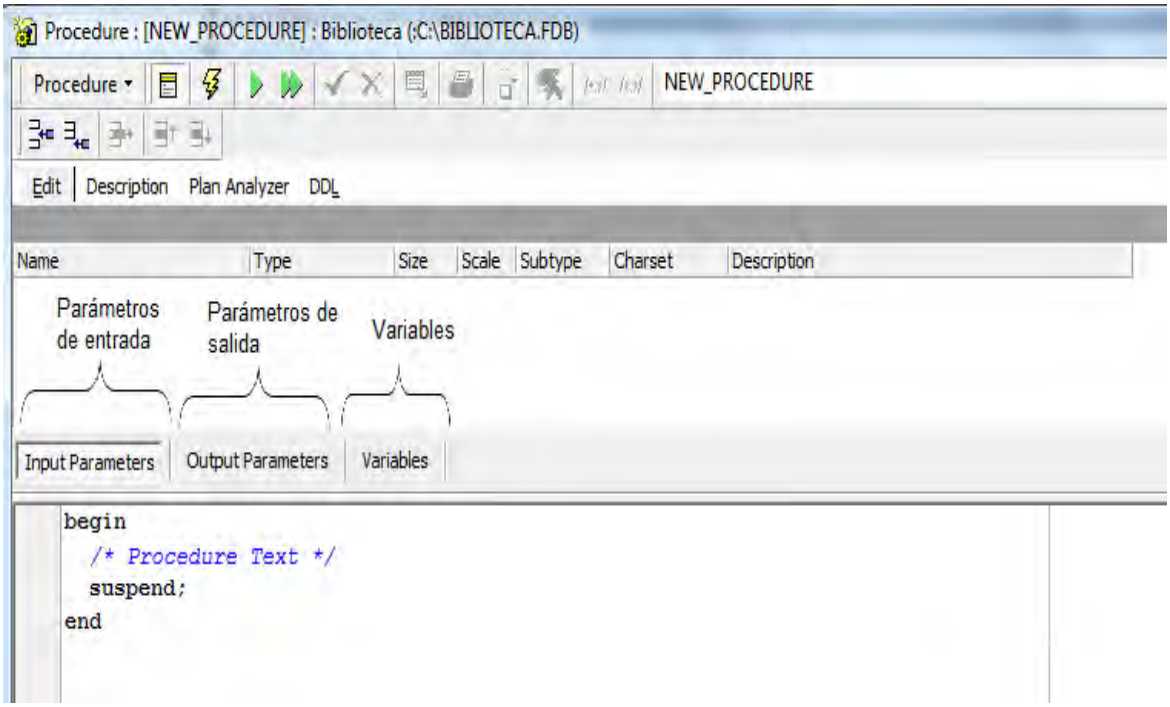


Figura 4.3.23 Ventana para la creación de un Stored Procedure

Una vez que el Stored fue escrito se verifica el código y si es correcto entonces se agrega en la carpeta de “Procedures” del menú izquierdo. Figura 4.3.24.

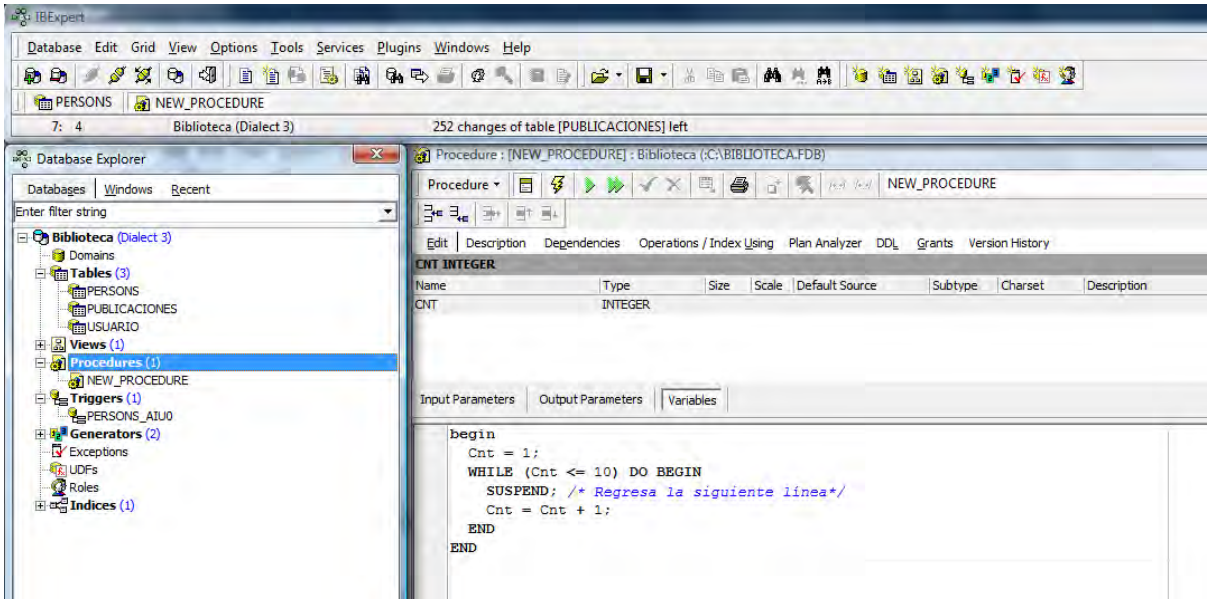


Figura 4.3.24 Script de la creación del Procedimiento

El sistema cuenta con la definición de 8 procedimientos almacenados los cuales se muestran en la siguiente figura 4.3.25 en la ventana izquierda se muestra el árbol de procedimientos almacenados, en la ventana derecha se puede observar la definición de uno de los procedimientos.

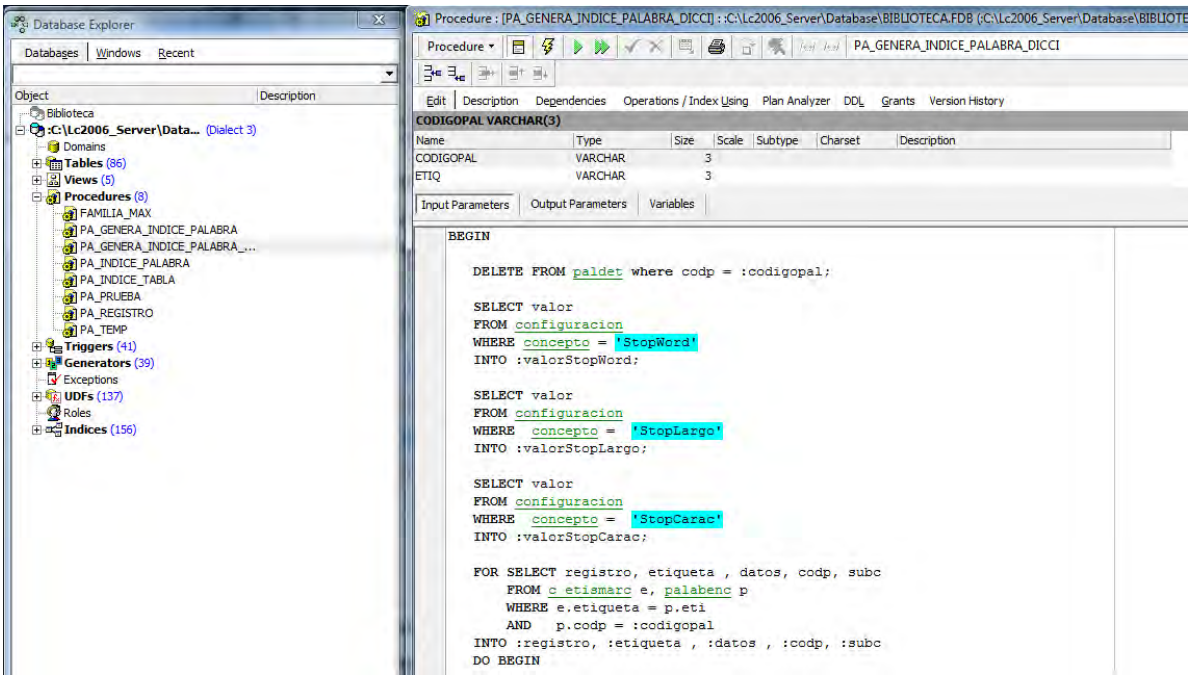


Figura 4.3.25 Árbol de procedimientos almacenados del Sistema

Respaldos de la base de datos (Backup)

La distribución de Firebird contiene una utilidad para realizar respaldos y restaurar bases de datos, su nombre es *gbak* y se encuentra dentro del directorio /bin. Las bases de datos se pueden respaldar mientras hay usuarios conectados al sistema haciendo el trabajo normal, el respaldo es tomado en un instante de la base al comienzo del mismo, es recomendable que los respaldos y las ocasionales restauraciones usando gbak fueran tareas programadas en las actividades administrativas de las bases de datos. La versión 1.5 estable de Firebird soporta respaldos totales a una base de datos sin embargo no soporta respaldos incrementales.

Para realizar respaldos se utilizará el comando gbak, <origen> que será la base de datos a la cual se desea realizar el respaldo, <destino> será el nombre del archivo del respaldo, la extensión usual del archivo de respaldo es .fbk para Firebird. Sólo el usuario SYSDBA o el propietario de la base pueden realizar respaldos de la misma.

Sintaxis general.

gbak <opciones> -user <nombre_usuario> -password <contraseña> <origen> <destino>

En la figura 4.3.26 se muestran las opciones generales para el comando gbak.

-nodbtriggers	Suprime los Triggers en la base de datos
-pas [sword] <contraseña>	Contraseña de la base de datos
-role <rol>	Conectar a la base con algún rol
-se [rvice] <hostname>:service_mgr	Respaldo: Crea un archivo de respaldo en el servidor de base de datos. Restauración: Crea una base de datos con el archivo de respaldo en el servidor
-u [ser] <nombre_usuario>	Nombre del usuario de la base de datos
-v [erbose]	Comenta todos los procesos que realiza gbak
-y	Modo callado
-z	Muestra la versión de GBAK

Figura 4.3.26 Opciones generales del comando GBAK

La siguiente figura 4.3.27 muestra las opciones de GBAK para realizar un respaldo.

-b [ackup_database]	Respaldo. Opcional
-co [nvert]	Convierte tablas externas en tablas internas
-e [xpand]	Crea un respaldo que no puede ser comprimido
-fa [ctor] n	Factor de bloque para algún dispositivo de cinta.
-g [arbage collect]	No realiza el garbage collection (barrido) durante el respaldo
-ig [nore]	Ignora los errores de comprobación durante el respaldo

-l [imbo]	Ignora las transacciones pendientes durante el respaldo
-m [etadata]	Solo respalda los metadatos (esquema). Las tablas creadas por el usuario no serán guardadas.
-nt	Formato no transportable (usar solo cuando se requiera restaurar en la misma plataforma y versión de la base de datos)
-t [ransportable]	Crea una versión transportable entre plataformas y versiones de servidores por default.

Figura 4.3.27 Opciones para respaldo del comando GBAK

Se realizará un respaldo “normal” de la base de datos que se creó anteriormente llamada biblioteca.fdb. (Figura 4.3.28).

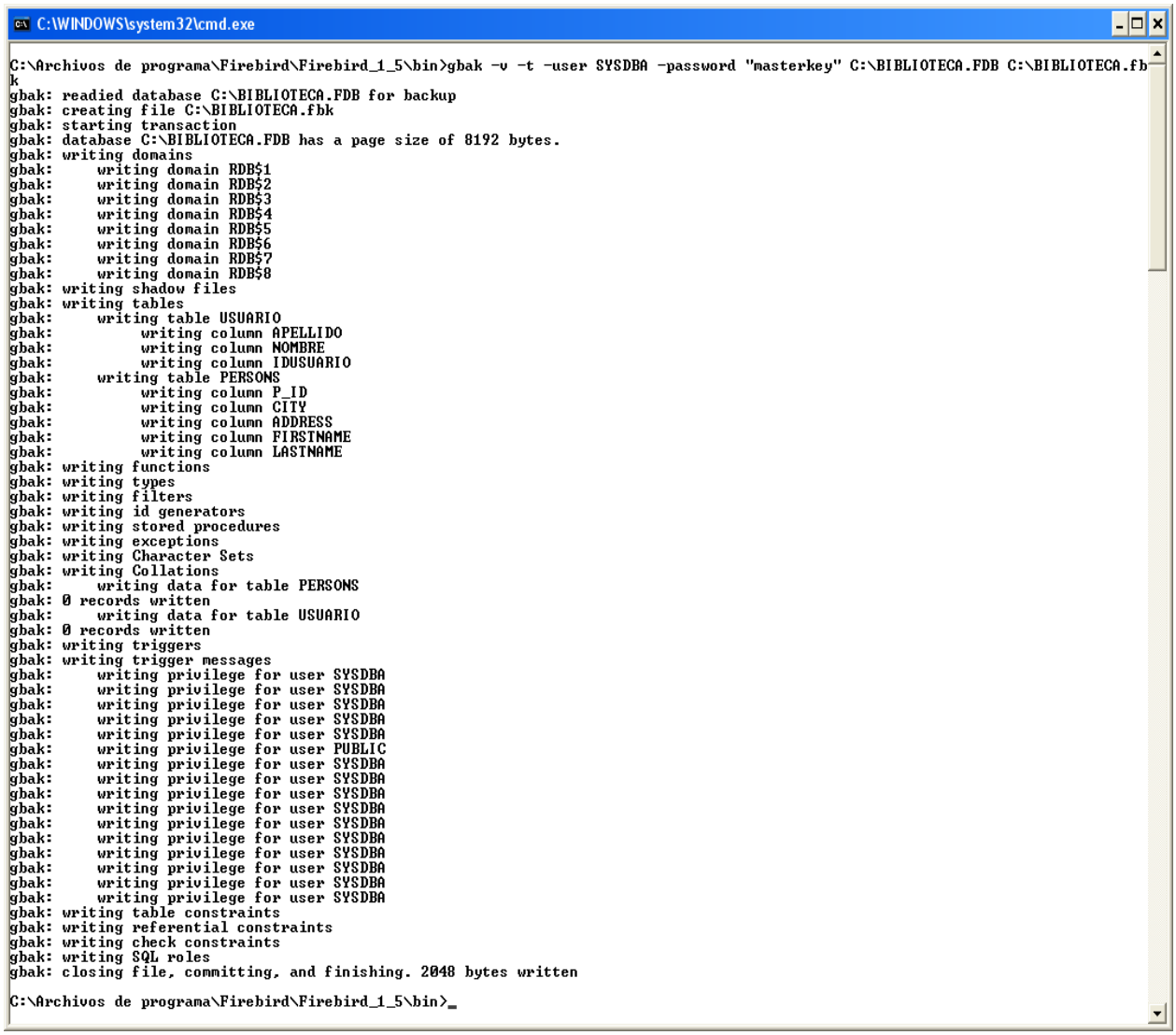


Figura 4.3.28 Respaldo de la base de datos Biblioteca

Restauración de la base de datos respaldada

Para restaurar una base de datos se utiliza el mismo comando GBAK, sin embargo, los parámetros de entrada son diferentes, en la figura 4.3.29 se muestran dichos parámetros.

-bu [ffers]	Ajusta el tamaño del cache para la base de datos restaurada
-c [reate_database]	Todos los índices serán restaurados como INACTIVOS
-k [ill]	No crea sombras que son definidas durante el proceso de respaldo
-mo [de] read_write	Restaura una base de datos de escritura/lectura por default.
-mo [de] read_only	Restaura una base de datos de solo lectura
-n [o_validity]	No restaura los constrains, así que una vez restaurada la base los constrains entre las bases deberán ser restaurados de otra manera
-p [age_size] <tamaño>	Ajusta el tamaño de página de la nueva base, puede ser 1024, 2048, 4096, 8192, por default es 1024
-r [eplace_database]	Restaura sobre una base de datos existente, esto sólo puede ser realizado por el SYSDBA o el propietario de la base, misma que es sobrescrita.
-use_[all_space]	Normalmente en las restauraciones, las páginas de la base de datos son llenadas

	cerca del 80%, con la opción use_all_space, las páginas de la base de datos será llenada al 100%. Práctico cuando se restauran bases de sólo lectura.

Figura 4.3.29 Opciones para la restauración del comando GBAK

Se realizará una restauración normal de la base de datos que respaldamos anteriormente llamada Biblioteca.fbk. (Figura 4.3.30).

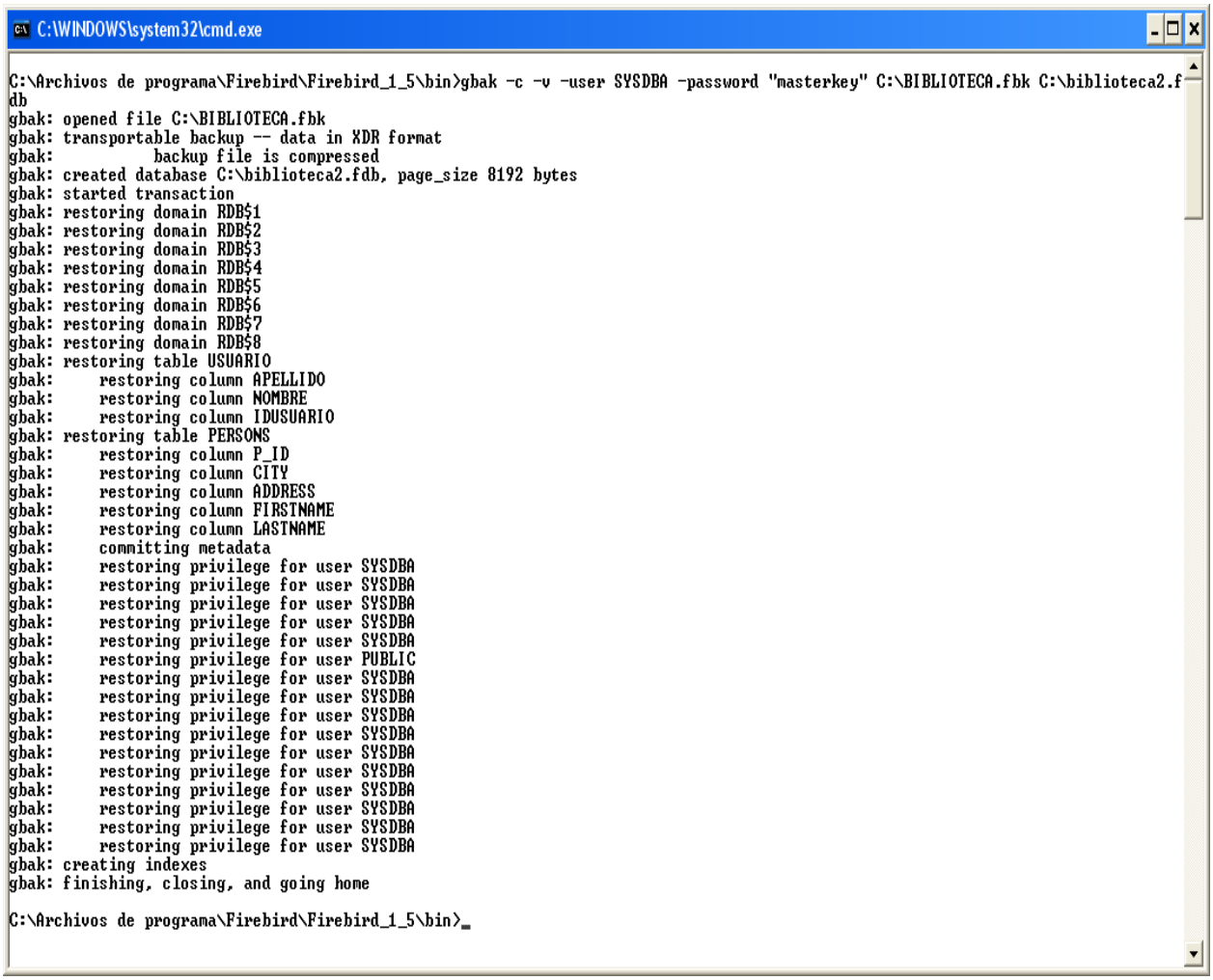


Figura 4.3.30 Restauración de la base de datos Biblioteca

Para restaurar una base de datos existente el comando cambiará de acuerdo a las opciones mencionadas en éste caso restauraremos la base de datos de Biblioteca (figura 4.3.31).

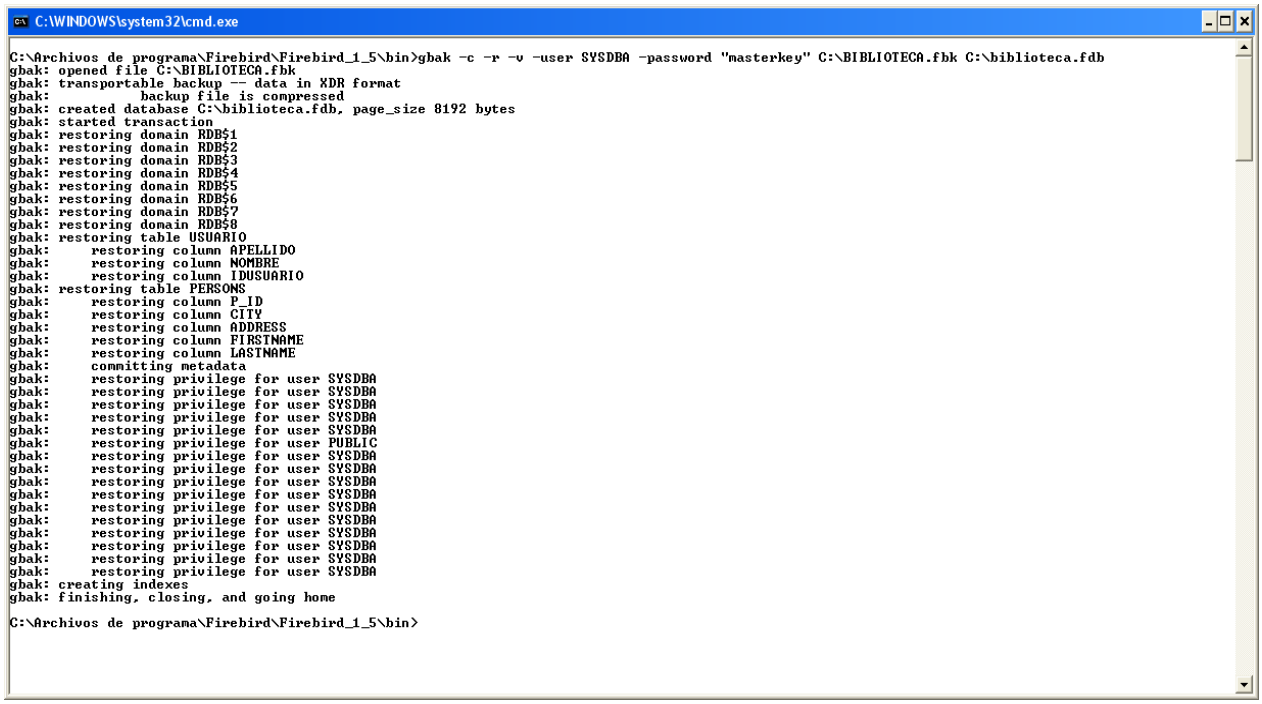


Figura 4.3.31 Restauración de la base de datos Biblioteca sobre la base Biblioteca en uso

Restaurando un respaldo de sólo lectura de la base de datos Biblioteca. Figura 4.3.32

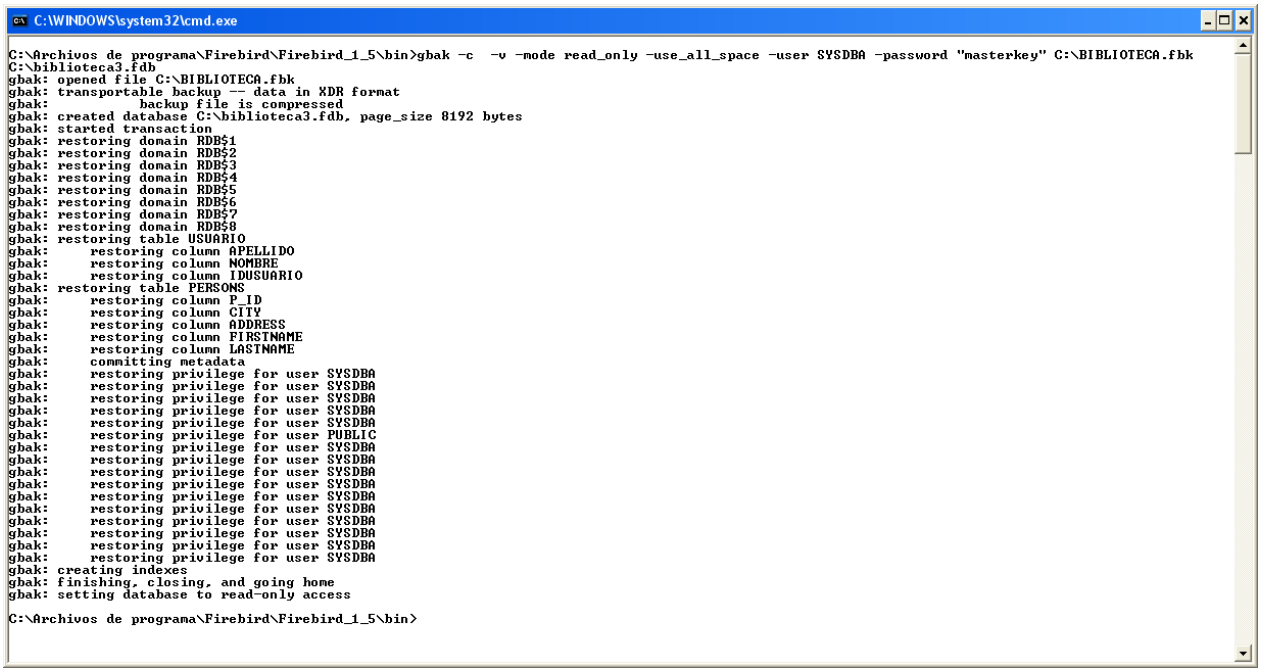


Figura 4.3.32 Restauración de la base de datos Biblioteca de solo lectura

4.4 DISEÑO Y CONSTRUCCIÓN DEL FRONT – END

El sistema es un software para la automatización de bibliotecas como se muestra en la figura 4.4.1 integra los módulos que apoyan y controlan los procesos de gestión de bibliotecas.

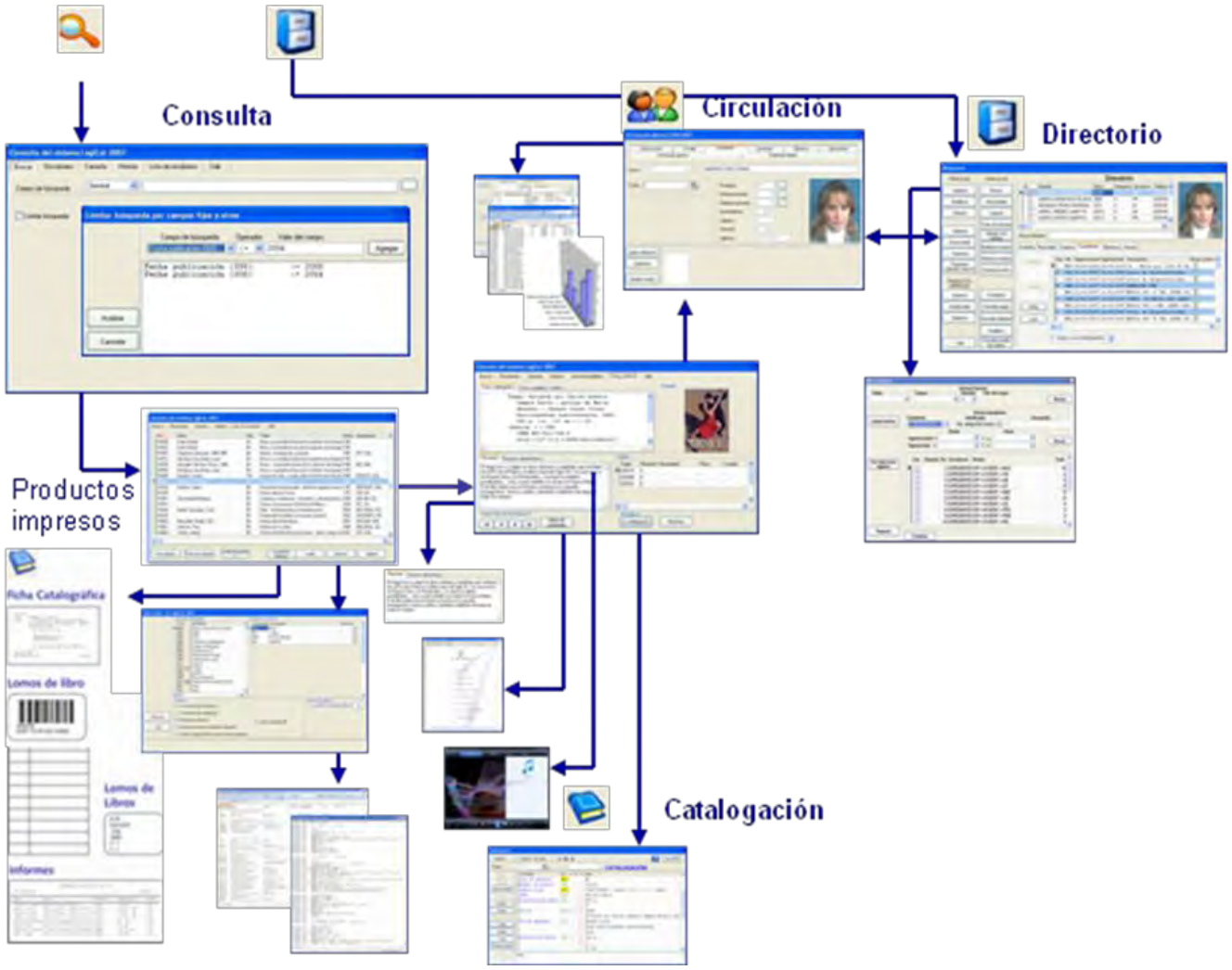


Figura 4.4.1 Diagrama general del Front-End

Navegación de consulta

Como se muestra en la figura 4.4.2 la pantalla de consulta fue diseñada pensando en una búsqueda rápida y sencilla para el usuario, se decidió primero poner un combo en el cual se puede seleccionar el tema a buscar esto porque así se pueden dividir las consultas, después se puso un componente de texto en el cual se pone el texto que se quiere buscar, un botón para iniciar la búsqueda y en cuanto la búsqueda termina muestra el resultado en un componente grid como se muestra en la figura 4.4.3.

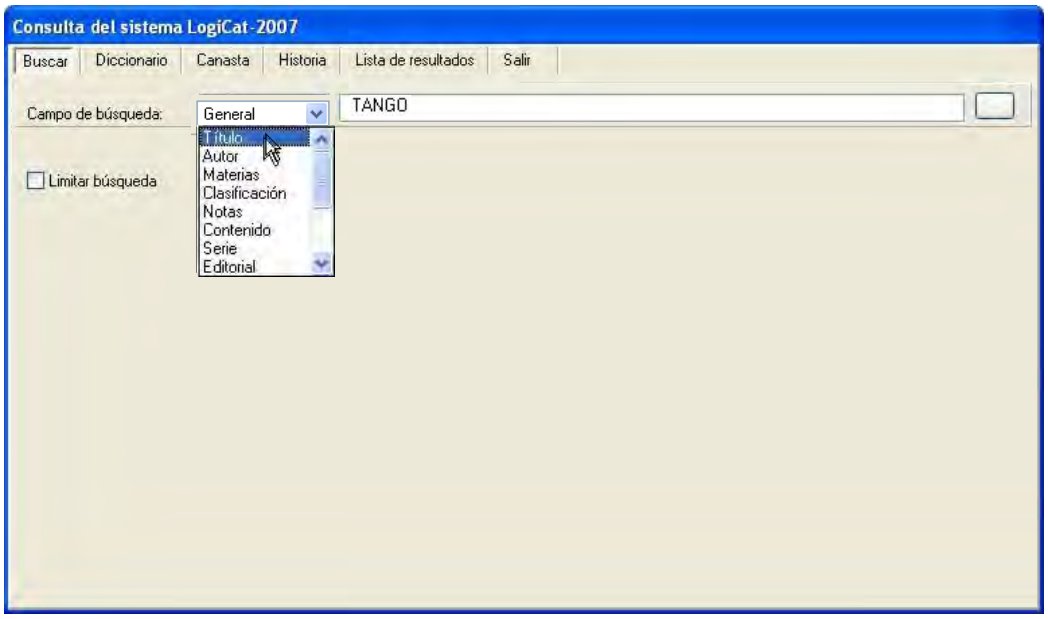


Figura 4.4.2 Pantalla de consulta

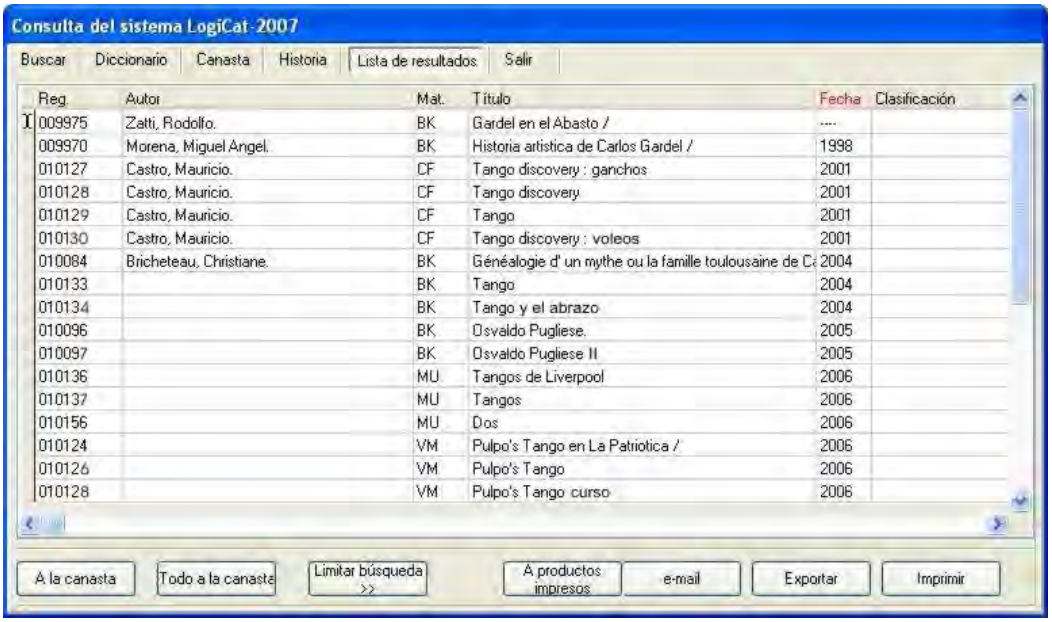


Figura 4.4.3 Pantalla de listado de resultados

Ya que se muestra la información de la búsqueda se puso un menú donde se pueden tener varias opciones.

Una opción es la de ficha como se muestra en la figura 4.4.4 donde se decidió poner la información del registro buscado con un panel el cual incluye la ficha catalográfica un componente imagen donde se puede poner una portada, otro panel para un resumen y un grid para las copias que tiene ese materia.

Figura 4.4.4 Pantalla de ficha

Navegación de catalogación

Como se muestra en la figura 4.4.5 la pantalla de catalogación consulta fue diseñada pensando en una catalogación rápida y sencilla para el usuario, se decidió poner un conjunto de botones del lado izquierdo los cuales le permiten al usuario una fácil selección de las diferentes opciones, se colocó un componente de texto en el cual se pone el número de registro que se quiere buscar, se colocó un componente grid el cuál muestra la información del registro la figura 4.4.6 muestra la información de un registro.

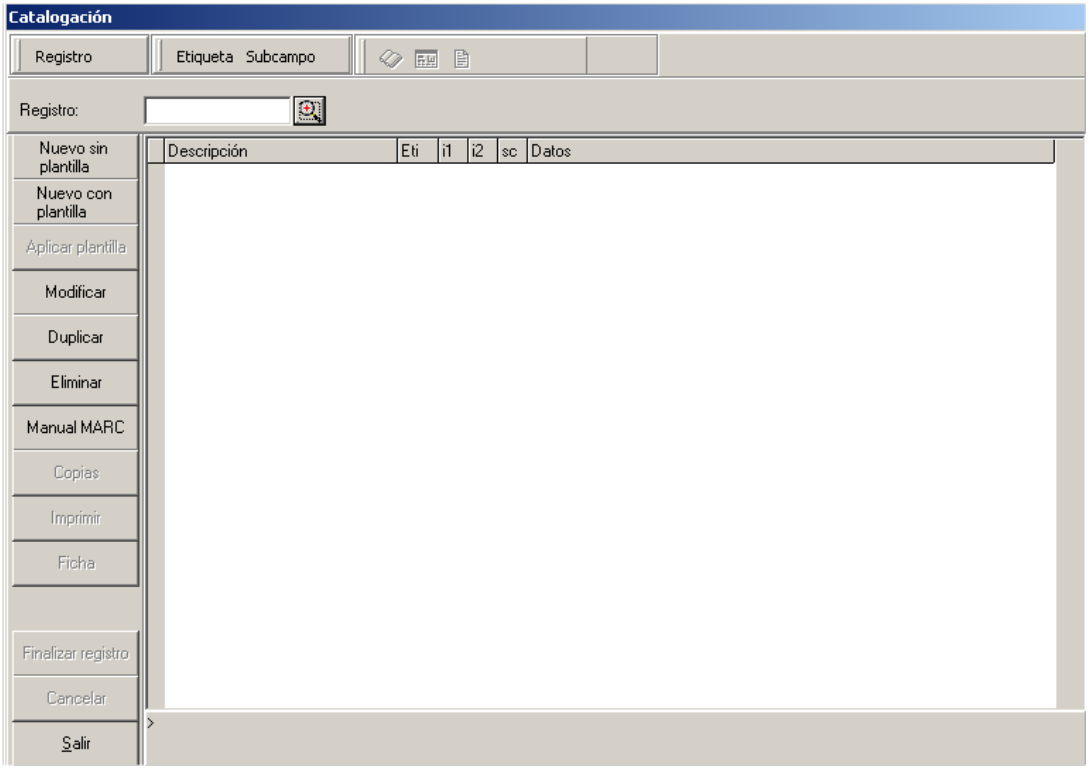


Figura 4.4.5 Pantalla de módulo de catalogación

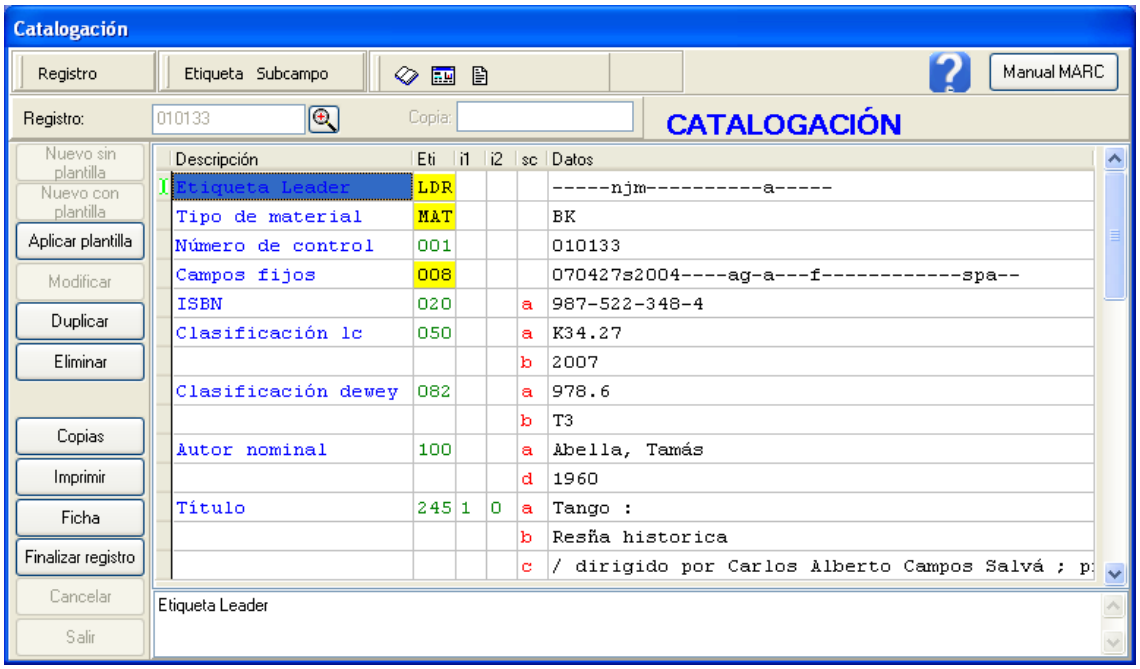


Figura 4.4.6 Pantalla de catalogación con plantilla

Navegación de circulación

La pantalla de circulación como se muestra en la figura 4.4.7 se colocaron dos componentes de texto para buscar al alumno y el material de una manera fácil y sencilla en la parte de arriba se colocó un menú con los diferentes tipos de opciones que hay en el modulo de circulación, también se colocó una serie de componentes para mostrar la información del usuario de biblioteca así como su fotografía en un componente de imagen, en la parte de abajo se colocó un componente grid para ir mostrando las transacciones que se van haciendo.

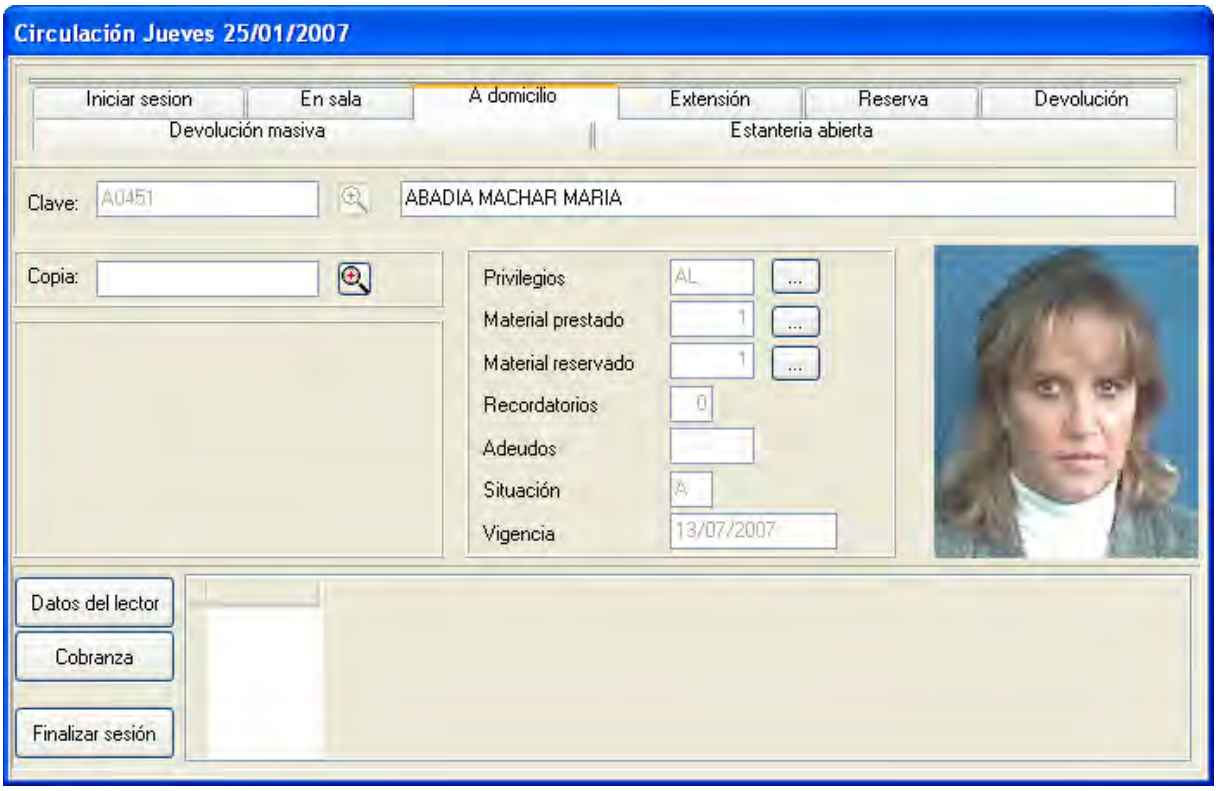


Figura 4.4.7 Pantalla de circulación

Navegación de directorio

Contiene información de personas e instituciones relacionadas con la biblioteca y los procesos que realiza como se muestra en la figura 4.4.8.

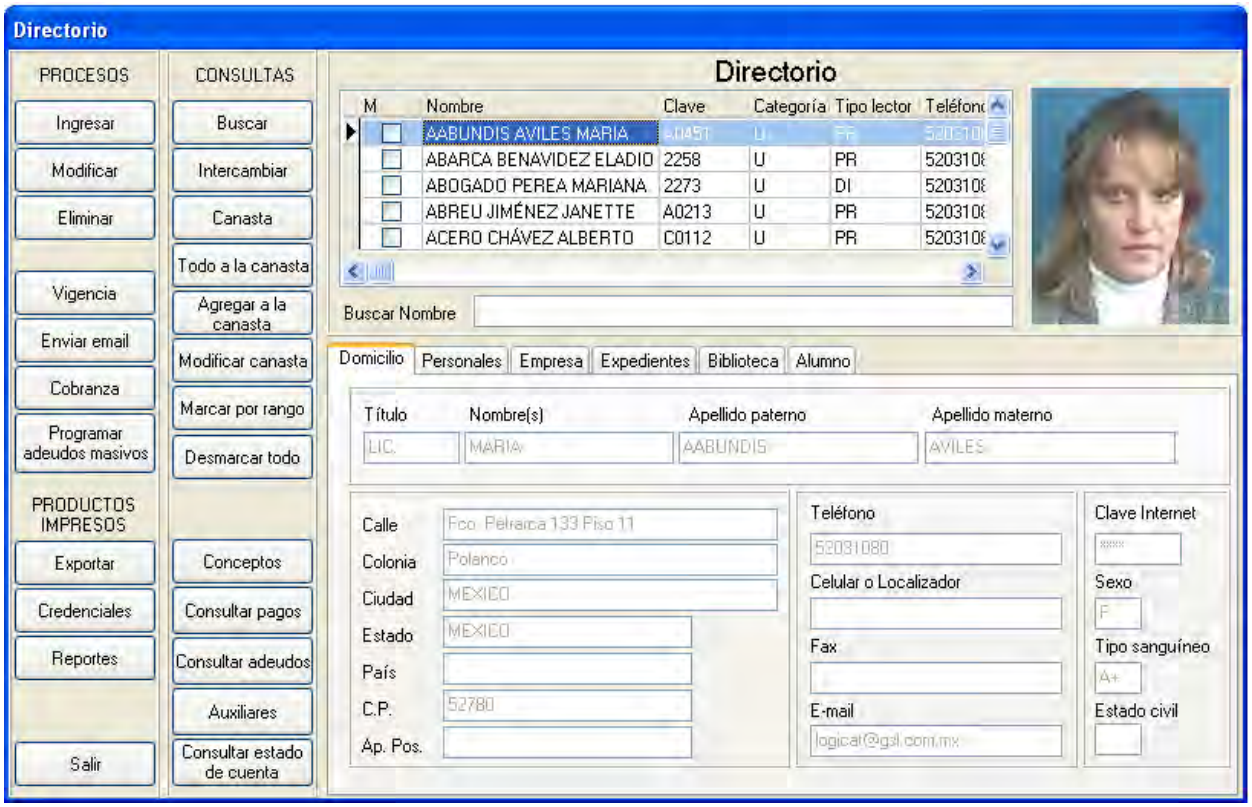


Figura 4.4.8 Pantalla de directorio

Se decidió poner del lado izquierdo dos conjuntos de botones para facilitar al usuario la selección de las diferentes opciones como: ingresar, modificar, eliminar, exportar etc. En la parte del superior central se colocó un componente grid en el cuál se muestra la información básica del alumno en la parte del centro abajo se colocó un panel con diferentes pestañas que muestran la información detallada del usuario de la biblioteca, en la parte de arriba derecha se colocó un componente imagen para poner la foto.

4.5 INTEGRACIÓN Y PRUEBAS DE SISTEMA

La integración y pruebas del sistema es una de las últimas fases del desarrollo de software, consiste en analizar el producto de software previo a ser liberado en un ambiente de producción, por lo tanto se realizan diversos procesos con el fin de revelar la calidad del producto, usabilidad, encontrar defectos y establecer mejoras, además de verificar el grado de cumplimiento respecto a las especificaciones iniciales.

Por lo general, los desarrolladores hacen una diferencia entre los errores de programación o también conocidos como “bugs" y los defectos de forma. En un defecto de forma, el programa no realiza lo que el usuario final espera, un error de programación se define como un fallo en la semántica de un programa, sin embargo el error de programación podría presentarse como un defecto de forma.

Es recomendable que el proceso de pruebas sea realizado por un grupo independiente a los que colaboraron dentro del desarrollo del producto de software, con el fin de evitar verificaciones y pruebas someras, así como para corroborar que el sistema ha sido interpretado correctamente, el grupo que realiza las pruebas es conocido como “testers”.

Una práctica que viene popularizándose es la distribución de forma gratuita de una versión final del producto para que los propios consumidores sean los que realicen las pruebas, para estos casos la versión del producto se le denomina beta y a dicha versión de pruebas beta testing, a la versión final del producto se le conoce como versión final o master. La fase de pruebas puede evidenciar la presencia de errores más no la ausencia de ellos.

Para la fase de pruebas se trata de someter al producto de software a pruebas exhaustivas en todas las situaciones posibles con el fin de encontrar algún fallo, pero en

productos grandes de software es una tarea difícil, pues el número de módulos, métodos, atributos y líneas son finitos, sin embargo el número de ciclos que podrían tomar, lleva a una infinidad de posibles caminos a seguir. Por lo tanto el número de combinaciones se incrementa a tal grado que resulta prácticamente imposible ejecutar y analizar todos los casos posibles que un usuario pudiera realizar en un tiempo determinado.

A pesar de estas limitaciones, las pruebas de sistemas son una parte integral dentro del desarrollo de software, usualmente más del 50% del tiempo de desarrollo se utiliza en pruebas, no es posible probar la calidad directamente, pero pueden relacionarse factores para hacer visible la calidad de un producto de software. La calidad de un producto de software puede ser dividido en tres grupos, funcionalidad, ingeniería y adaptabilidad. Estos factores pueden concebirse como dimensiones dentro del espacio de calidad del software. Cada dimensión puede ser separada en componentes y consideraciones de más bajo nivel. En la figura 4.5.1 se muestran algunas de las consideraciones de calidad más citadas.

Funcionalidad (Calidad Exterior)	**Ingeniería (Calidad interior)**	**Adaptabilidad (Calidad futura)**
Fiable	Eficiente	Flexible
Usable	Probado	Reusable
Integrable	Documentado	Mentenible

Figura 4.5.1 Factores típicos de calidad del software

La importancia de uno u otro factor varía de aplicación a aplicación. Por ejemplo sistemas donde vidas humanas están en juego, se debe poner especial énfasis en los factores fiables e íntegros, en un sistema de negocios típico el uso y mantenimiento deberán ser los factores clave. Para que sean eficaces las pruebas deben estar

orientadas a la medición de cada factor, obligándolas a convertirse en pruebas tangibles y visibles.

Pruebas que tienen como propósito el validar que el producto de software funcione correctamente son llamadas pruebas limpias o pruebas positivas, la desventaja es que sólo pueden validar el producto específicamente para los casos de uso. Un número finito de pruebas no puede validar que el producto funcione correctamente para todas las situaciones. La presencia de al menos una falla en este tipo de pruebas, es suficiente para mostrar que el producto no funciona correctamente. Pruebas sucias o negativas se refieren a intentar hacer que el producto se corrompa, el producto debe tener la capacidad de manejar un nivel significativo de pruebas negativas.

En un proyecto de software deberán ser considerados las siguientes pruebas para garantizar un producto fiable:

- Pruebas de caja blanca

  Cobertura
  - De segmentos
  - De ramas
  - De condición / decisión
  - De bucles
- Pruebas de caja negra
  - Cobertura de requisitos
- Prueba de integración
- Prueba de regresión
- Pruebas de volumen.
- Pruebas de seguridad
- Pruebas de aceptación
- Pruebas alfa y beta
- Otras pruebas

Pruebas de caja blanca

También conocida como pruebas estructurales o de caja transparente, está basado en el conocimiento de la lógica interna del código de la aplicación. Las pruebas se aplican sobre las sentencias del código; segmentos, ramas y condiciones. Este tipo de pruebas la realiza el desarrollador muchas veces en conjunto o con ayuda de un usuario responsable del módulo del sistema, el desarrollador conoce la lógica del sistema y la estructura del componente para poder obtener los datos de prueba.

Las metas de las pruebas de caja blanca son las siguientes:

- Garantizar que todas las ramas independientes dentro de un módulo hayan sido ejecutadas dentro del mismo al menos una vez.
- Ejecutar todas las condiciones lógicas (verdaderas y falsas).
- Ejecutar todos los ciclos en sus límites operacionales.
- Ejecutar las estructuras de datos internas para asegurar su validez.

El total de pruebas de caja blanca se le conoce como “cobertura”, que es el número porcentual que indica el código del programa que ha sido probado.

Cobertura de segmentos

Igualmente conocida como “cobertura de sentencias, se define como una secuencia de sentencias sin puntos de decisión. El número de líneas de código en un programa es finito, es suficiente con tomar el código fuente e ir contando línea por línea. Es posible diseñar un plan de pruebas que vaya ejecutando más y más líneas hasta que se haya cubierto todas o al menos la mayoría. El proceso de pruebas finaliza antes de llegar al 100% ya que puede resultar excesivamente costoso y laborioso provocar que se pase por todas y cada una de las líneas del programa.

Cobertura de ramas

Cuando en el código del programa se presentan sentencias opcionales como condiciones de tipo “IF” o “SWITCH” es difícil tener una cobertura, pues dependiendo de los parámetros de entrada en la sentencia puede ejecutar un bloque de código u otro. Para hacer frente a estos casos se plantea un refinamiento de cobertura de segmentos y consiste en recorrer todas las posibles salidas de los puntos de decisión.

Cobertura de condición / decisión

La cobertura de decisiones prueba el número de decisiones ejecutadas, considerando que fue ejecutada una decisión cuando se han recorrido todas sus posibles ramas (la que la hace verdadera y la que la hace falsa, pero también todas sus posibles ramas en un switch).

Cobertura de bucles

Un bucle se ejecuta un número de veces, pero el número que fue ejecutado deberá ser muy preciso pues el hecho de ejecutarlo una de vez de más o una vez de menos puede traer problemas, por lo tanto se definen pruebas para cada tipo de bucle existente.

Bucles tipo while.

```
While<condicion> Then
        <bloque de código>
End
```

Prueba 1. Cero ejecuciones.

Prueba 2. Una ejecución.

Prueba 3. Más de una ejecución y hasta que la condición no se cumpla.

Bucles tipo Do – While

```
Do
        <bloque de código>
While<condicion>
```

Prueba 1. Primera ejecución.

Prueba 2. Más de una ejecución y hasta que la condición no se cumpla.
El ciclo for es más seguro pues en su misma sintaxis se define el número de veces que se ejecutará, y el compilador se encarga de garantizarlo. Sin embargo es recomendable analizar el contenido del bucle for pues puede darse el caso que dentro del mismo se altere alguna variable que se utilice en el cálculo del incremento o límite de iteración.

La ejecución de pruebas de caja blanca puede llevarse a cabo por medio de un depurador (que permite la ejecución paso a paso), una lista del módulo y un rotulador para ir marcando la línea que se va ejecutando, sin embargo, es una tarea muy tediosa y puede ser automatizada. En la actualidad existen compiladores que en el momento de generar el código de máquina dejan incrustado en el código instrucciones para que posteriormente a una ejecución indique el número de veces que se ha ejecutado cada sentencia, rama, bucle, etc.

El tener una buena cobertura en pruebas de caja blanca es un objetivo muy deseable, pero no suficiente, pues un programa puede estar muy bien en su código, sin embargo no tener la funcionalidad por el que fue creado.

Pruebas de caja negra

También conocidas como de caja opaca, funcionales, entrada/salida o inducidas por datos. Esta prueba se realiza con base a los requerimientos sin el conocimiento sobre cómo fue construido el sistema y por lo general dirigido a los datos, se enfoca directamente sobre el exterior del módulo o componente sin importar el código. Por lo tanto su comportamiento sólo puede ser dado al estudiar sus entradas y sus salidas relacionadas a éstas.

Como se mencionó anteriormente el software no debe ser probado por el creador o grupo de creadores del sistema pues el conocimiento de la estructura interna del programa limita la variedad de datos que pueden ser probados o el encaminamiento de

las pruebas es hacia puntos específicos, ejercitando partes del software olvidadas por los desarrolladores por su simpleza en la creación. Por lo tanto es aspecto humano es esencial en las pruebas de caja negra pues se aplican sucesos de la vida real a la prueba, como pueden ser errores de tipo, o trabajar equivocadamente sobre la aplicación, etc., pero los desarrolladores muchas veces pasan por alto este tipo de errores.

Las pruebas de caja negra trata al programa como una función matemática, en donde hace el estudio de las respuestas o salidas que pueden modelarse como el "codominio" de los datos entrantes "dominio". Además éste tipo de pruebas tiene otras metas, determinar la eficiencia del programa desde el desempeño en el equipo, el tiempo de retardo de las salidas hasta el nivel de recuperación del sistema luego de provocar alguna falla o caída producidas por manejo incorrecto de datos, equipo o producidas por cortes de energía.

Pruebas de integración

Se llevan a cabo durante la construcción del sistema, se involucra un número creciente de módulos y se finaliza probando el sistema en conjunto. Se pueden plantear desde dos puntos de vista; estructural o funcional. Las pruebas estructurales son parecidas a las pruebas de caja blanca, pero se trabaja a un nivel conceptual superior. Pues en lugar de referirse a sentencias de lenguaje, se refiere a llamadas entre módulos. Por lo tanto trata de identificar todos los posibles esquemas de llamadas y ejercitarlos para lograr una buena cobertura de segmentos o de ramas. Por otra parte las pruebas funcionales son parecidas a las pruebas de caja negra. Pues trata de encontrar fallos en las respuestas de un módulo cuando su operación depende de las respuestas de otros módulos.

Las pruebas finales cubre el sistema completo además de que pretende cubrir plenamente la especificación de requisitos del usuario. Además, para este grado en el

desarrollo del proyecto, se debe tener el manual de usuario, que igualmente se utiliza para realizar pruebas hasta lograr una cobertura aceptable.

Pruebas de regresión

Cada vez que se agrega un nuevo módulo como parte de las pruebas de integración, el software cambia, por lo tanto se establecen nuevos caminos en el flujo de datos y pueden existir nuevas I/O y se invoca una nueva lógica de control, con los cuales pueden existir problemas con funciones que ya trabajan correctamente. Por lo tanto, las pruebas de regresión consisten en volver a ejecutar un subconjunto de pruebas que se han llevado a cabo anteriormente para verificar que los cambios por agregar un módulo no tienen efectos colaterales indeseados.

Pruebas de volumen

Las pruebas de volumen se realizan para verificar el funcionamiento adecuado y eficiente del producto bajo condiciones extremas de operación, en procesos continuos y paralelos, comprobando la inexistencia de un mal funcionamiento.

Pruebas de seguridad

Las pruebas de seguridad tratan de verificar que los mecanismos de protección del sistema sean adecuados y así evitar entradas inválidas. Cuando se realiza éste tipo de prueba el encargado funge como un individuo que desea vulnerar la seguridad del producto.

Pruebas de aceptación

Las pruebas de aceptación se realizan junto con el cliente, y se analiza la funcionalidad del mismo. Por lo tanto el cliente que en un principio acordó los requerimientos junto con el equipo de desarrollo, verifica que cada uno de los procesos y casos de uso funcionen como se propuso. Antes de realizar la prueba de aceptación se debieron de

haber realizado las pruebas anteriores para evitarse errores de la aplicación frente al cliente.

Pruebas alfa y beta

Una vez que las pruebas de aceptación fueron realizadas de manera exitosa, el equipo de desarrollo promueve las pruebas “alfa” y “beta”, la prueba alfa consiste en invitar al cliente a que trabaje con el producto. Se mantiene un sistema controlado y el cliente siempre cuenta con un experto el cual lo auxilia a usar el sistema, analizar los resultados y en caso de que un incidente sea registrado, saber manejarlo.

Las pruebas beta son posteriores a las pruebas alfa, y se desarrollan en el entorno del cliente, un entorno que se encuentra fuera de control. Para estas pruebas el cliente trabaja solo con el producto y trata de encontrar fallos y errores de los cuales hace un informe al desarrollador.

Este tipo de pruebas son muy comunes en productos que serán vendidos a muchos usuarios, algunos de los potenciales compradores les interesa participar en este tipo de pruebas, muchas veces el equipo de desarrollo ofrece algunos beneficios para los clientes que deseen participar en las pruebas beta.

Otras pruebas de software

- **Pruebas aleatorias.** Se basa en que la probabilidad de encontrar errores es la misma si se hacen una serie de pruebas aleatoriamente elegidas, que si se hacen siguiendo las instrucciones que dictan los criterios de cobertura de caja negra o blanca.

  El hecho de tomar módulos y procesos aleatoriamente en busca de errores pondrá en manifiesto los errores más patentes, sin embargo, dejará en oculto muchos errores potenciales que pudieran existir, por ello se debe hacer un balance, si la aplicación es poco crítica, puede que sea suficiente realizar

pruebas en módulos aleatoria, pero si es una aplicación muy crítica no se debe de tomar este criterio.

- **Pruebas de prestaciones.** Es importante tener en cuenta el tiempo de respuesta en cada uno de los módulos, dependiendo del número de datos a procesar, además de verificar el consumo de memoria, el espacio de disco, los datos que transfiere por el canal de comunicaciones, etc.
- **Prueba de Mutación.** Consiste en alterar ligeramente el sistema bajo las pruebas para introducir errores y verificar si las pruebas que se han realizado están verificando la aplicación correctamente.

Pruebas aplicadas al sistema

Es muy importante aplicar las pruebas al sistema en busca de errores ya sea de programación o del flujo y manejo de información, sin embargo es una tarea cansada y un poco engorrosa.

Es difícil estar revisando el código línea por línea para verificar el flujo de información, sin embargo se trataron de verificar todos los módulos, lo que se realizó con especial énfasis fue la cobertura de bucles pues si no son manejados correctamente pueden desestabilizar la aplicación totalmente.

En cuanto a las pruebas de caja negra, se realizaron los siguientes procesos:
Para la prueba de seguridad se trató de ingresar al sistema intentando firmarse con alguna cuenta no válida o mandando cadenas muy largas de caracteres los mensajes que el sistema nos enviaba fueron los correctos. Figura 4.5.2.

Figura 4.5.2 Prueba de validación del sistema

Para las pruebas de integración se revisó módulo por módulo tratando de encontrar errores en las capturas o validaciones. En el módulo de consulta se trató de ingresar caracteres no válidos, vacíos o muy largos para tratar de desestabilizar la aplicación, sin embargo en todos los casos se comportó adecuadamente y mandó los mensajes de error. Figura 4.5.3.

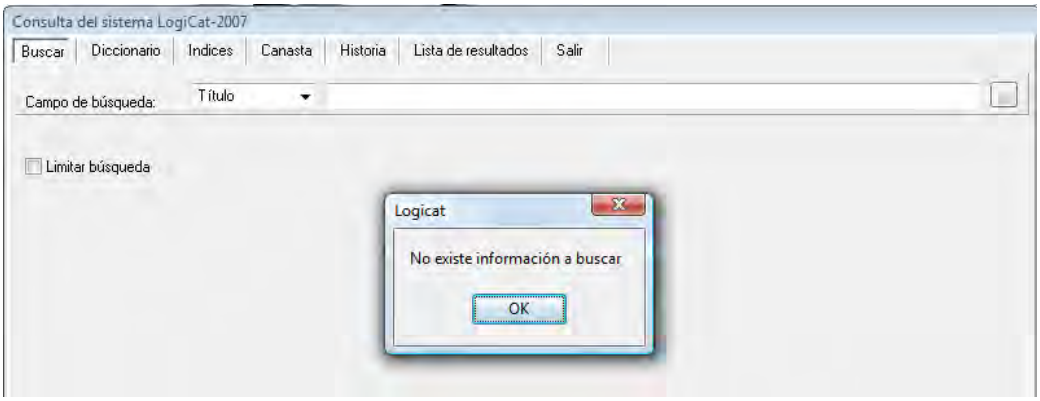


Figura 4.5.3 Pruebas en el módulo de consulta

Se realizaron pruebas de validación en los datos e ingresando datos incorrectos en la administración del directorio, y la aplicación manejó los errores de manera correcta. Figura 4.5.4.

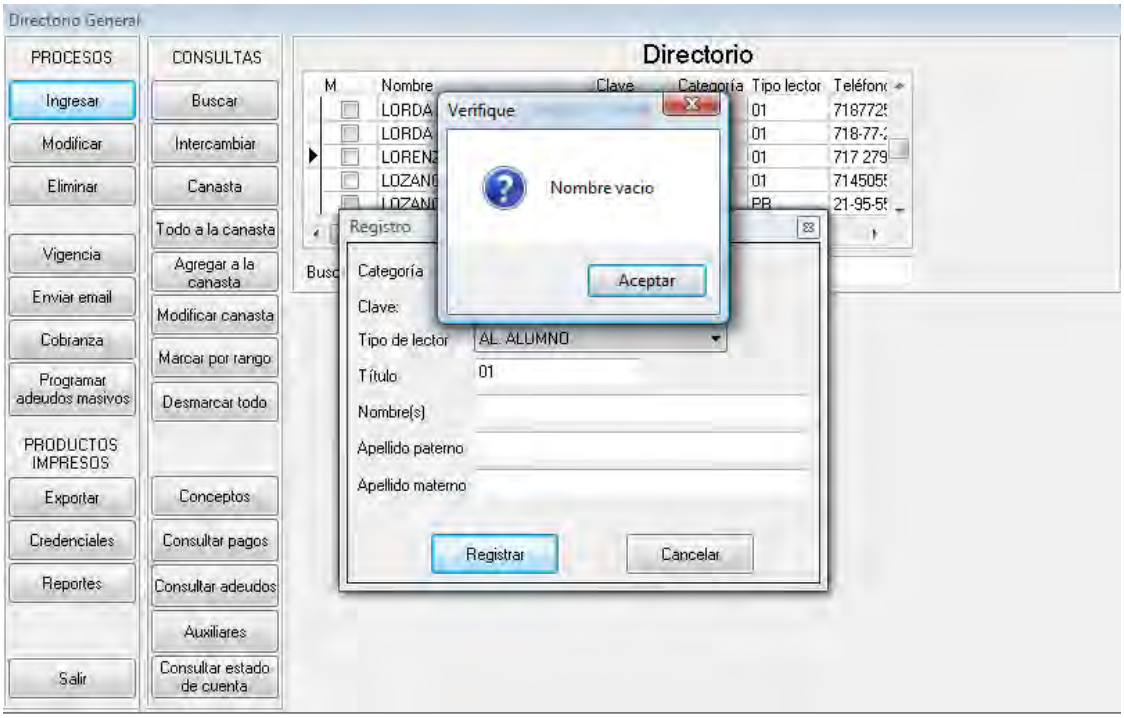


Figura 4.5.4 Validación en el directorio

Para realizar pruebas de volumen se abrieron varias aplicaciones en la misma máquina y en diferentes máquinas que comparten el mismo segmento de red y se trataron de realizar procesos en todas las aplicaciones y en ninguna nos envío un error de la base de datos o de recursos. Figura 4.5.5.

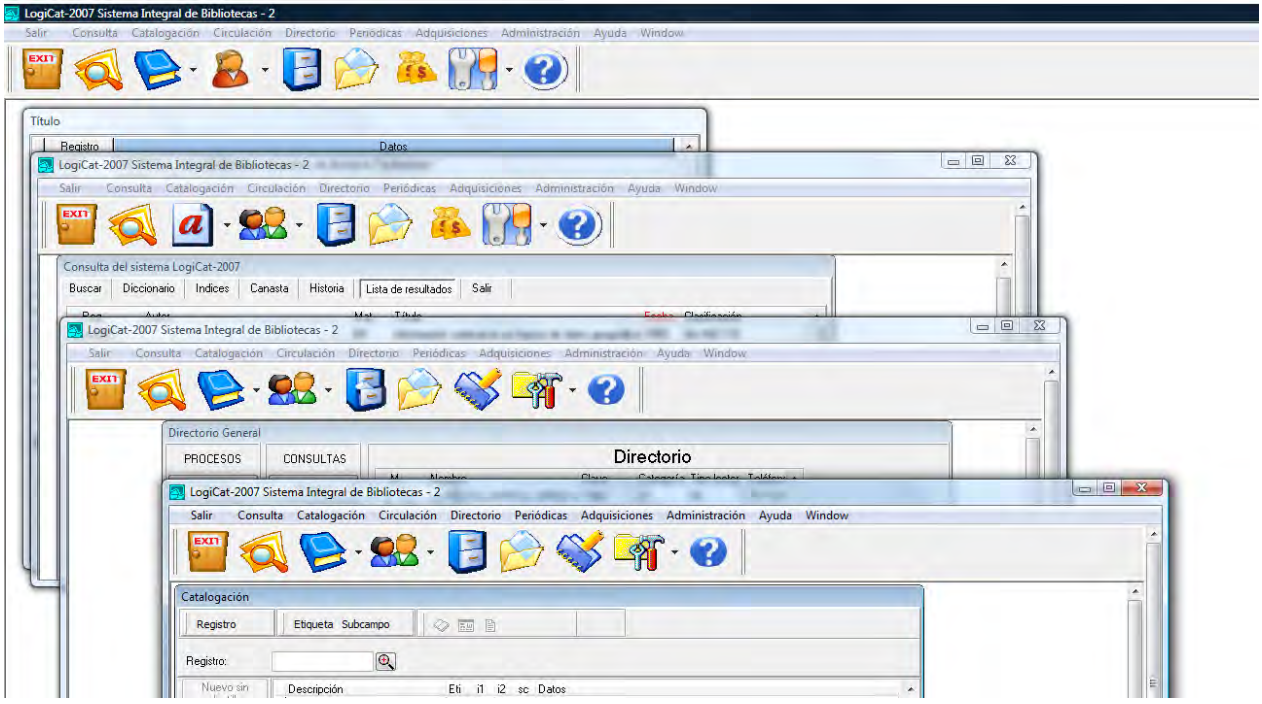


Figura 4.5.5 Prueba de volumen

Una vez que se estuvieron validando y probando cada uno de los módulos entonces se realizó la prueba de validación, confirmando que excedía de las expectativas siendo aprobada de esta manera.

Por lo tanto se dispuso a generar una versión que sería la prueba alfa para probarla en un ambiente controlado, en este caso en una copia de la base de datos de la biblioteca, pudiendo eliminar y agregar registros con el fin de familiarizarse con la aplicación y encontrar fallos en la lógica y en los procesos.

CONCLUSIONES

Se cubrieron las expectativas del sistema que se propusieron en un inicio con éxito ya que el sistema es capaz de gestionar totalmente el sistema bibliotecario pues puede realizar consultas, actualizar la base de datos de usuarios, monitorear sus movimientos y generar reportes para el personal que administra la biblioteca.

El uso de la tecnología en la automatización de procesos es un excelente medio para reducir tiempos de ejecución de tareas, márgenes de error e incrementar la productividad en lugares donde se requiere dar atención personalizada.

No importando si una tarea se ha ejecutado exitosamente muchas veces, siempre dicho proceso es candidato a ser automatizado y delinear su calidad para lograr la perfección.

El costo humano en la ejecución de una tarea con la posibilidad de ser automatizada será siempre más elevado que el costo para el desarrollo del sistema y/o la tecnología a crear.

El uso de bases de datos evita la necesidad de tener archivos físicos voluminosos, aumenta la velocidad para recuperar, registrar y actualizar datos, dando como resultado la obtención de información precisa y actualizada en cualquier momento.

Respecto a los resultados esperados, pudimos concluir que se cubrieron satisfactoriamente las expectativas de desempeño y desarrollo del sistema, ya que el seguir todos y cada uno de los puntos planteados al inicio nos llevó a consolidar un sistema completo, práctico y funcional, quedando listo para su implementación y contribuyendo con esto al desarrollo tecnológico.

El trabajo en equipo para la realización de este material fue muy enriquecedor y de una amplia retroalimentación, logrando una eficacia en el ritmo de trabajo y una participación total para la finalización de la tesis.

La importancia de conocer la problemática y la necesidad que tiene una institución o empresa, es relevante para la creación y desarrollo de un sistema que brinde una solución a dichas insuficiencias.

Los beneficios obtenidos repercuten en una reducción en el tiempo utilizado por los empleados para el registro de usuarios, la búsqueda de información por parte de los usuarios, la búsqueda de algún material, logrando con esto una mejora en la atención de usuarios y realizar un servicio de calidad.

Se fundamentó la solución del sistema al contar con una base de datos relacional que permite manejar la gran cantidad de información requerida, logrando contar con registros de movimientos de usuarios y haciendo más eficientes los procesos en la biblioteca.

El uso de tecnologías y estándares utilizados ampliamente en la actualidad no sólo fortalece la calidad del producto o servicio final, sino que facilita el mantenimiento y favorece la promoción del sistema para ser implementado en otras redes de bibliotecas o instituciones.

El caso de UML fue una herramienta útil para plasmar el análisis y diseño realizados, incluyendo las necesidades del sistema para la comprensión de elementos y alcances del mismo por los integrantes del equipo y facilita futuros mantenimientos o actualizaciones apegándose a estándares de documentación.

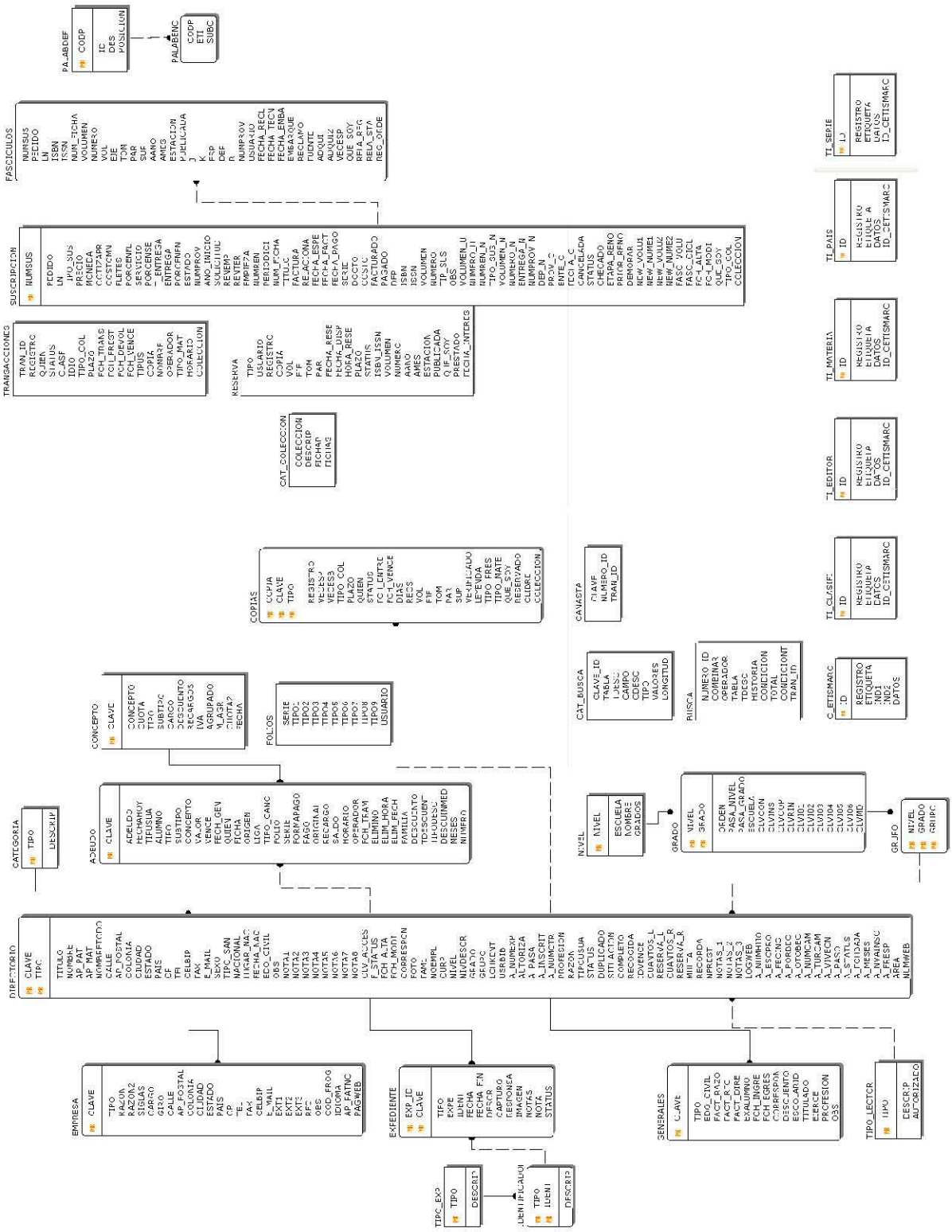
El sistema diseñado cumple con el objetivo de tener una herramienta de apoyo para agilizar los procesos administrativos, de gestión y técnicos de una biblioteca, logrando el apoyo a las funciones del personal que atiende a los usuarios desde la solicitud del material hasta su entrega.

Para realizar la automatización de cualquier servicio como un sistema de cómputo se requiere la existencia de un equilibrio entre el conocimiento y la experiencia en el desarrollo tecnológico, logrando un impacto positivo y eficaz en el servicio para la sociedad.

La evolución de los sistemas de gestión requiere el empleo de tecnologías cliente-servidor, permitiendo que la administración de datos abarque una gran variedad de tipos de información que puede estar disponible a cualquier hora.

El trabajo en equipo resultó muy apropiado y práctico, trabajando en forma colectiva e individual con base en nuestros perfiles, ámbitos y experiencia laboral, permitiendo aplicar los conocimientos adquiridos durante nuestra formación académica aunados a la experiencia de nuestro asesor de tesis.

APÉNDICE



BIBLIOGRAFÍA

Morales de Celestino, Elisa. Informática en la Biblioteca, una necesidad? Un reto. Lima 1988.

García Melero LA. Automatización de bibliotecas. Madrid: Arco/Libros. 1999

Jacquesson A. La información de las Bibliotecas: historia, estrategia y perspectivas.
París: Circulo de la Biblioteca. 1995.

Stalling, William. “Comunicaciones y redes de computadoras”. Prentice Hall, 6º Ed

Lankes, R. David, Kasowitz, A. S. “The AskA starter kit: how to build and maintain digital references services”, Syracuse University. NY, 1998.

Andrew S. Tanenbaum. Redes de Computadoras. Prentice Hall

James A. Senn. Análisis y Diseño de Sistemas de Información. Mc Graw – Hill.

Lucas Gómez Ángel. Diseño y Gestión de Sistemas de Bases de Datos. Parainfo.

Mark L. Gillenson. Administración de Bases de Datos. Limusa – Wiley, 2006.

Perry Greg. Aprendiendo Visual Basic 6 en 21 días. Prentice Hall, Pearson.

Ceballos, Francisco Javier. Curso de programación de Visual Basic 6. Alfaomega. 2000.

Kendal Kanneth E. Análisis y Diseño de Sistemas. Prentice Hall. 2005.

Larman, Craig. UML y patrones. Introducción al análisis y diseño orientado a objetos. Prentice Hall. 1999.

Charte, Francisco. Programación con Delphi. Anaya. 1996.

Charte, Francisco. Delphi 6 y Kylix. Anaya. 2001

Reisdorph, Kent. Aprendiendo Delphi. Prentice Hall. 1999.

Referencias electrónicas

http://es.wikipedia.org/wiki/Frame
http://www.astrossp.unam.mx/computo/red2m.jpg
http://sistemas.itlp.edu.mx/tutoriales
http://canalhanoi.iespana.es/informatica/redman
http://es.wikipedia.org/wiki/Arquitectura_de_red
http://www.arqhys.com/construccion/redes-arquitectura
http://es.wikipedia.org/wiki/Arquitectura_de_red
http://es.wikipedia.org/wiki/Clave_for%C3%A1nea