Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias de la Computación.

“MARCADORES SOCIALES Y FAVORITOS”

TESIS PROFESIONAL

Para obtener el Título de:

Ingeniero en Ciencias de la Computación.

Presenta:

VIQUE TEPANGO URZULO NAHUM.

Dirigida por:

M.C. Beatriz Beltran Martinez

Puebla, Puebla Junio 2008

INTRODUCCION 1

CAPITULO 1: MARCO TEÓRICO 3
1.1 INGENIERÍA DEL SOFTWARE 4
1.1.1 Definición Del Término Ingeniería De Software 4
1.2 RUP (Rational Unified Process o Proceso Unificado de Rational) Y UML (Unified Modeling Language o Lenguaje Unificado de Modelado) 5
1.2.1 Proceso Unificado de Rational 5
1.2.2 Características Esenciales 5
1.2.3 Ciclo de Vida 6
1.2.4 Que es UML 7
1.2.5 Modelos de UML 8
1.3 MODELOS DE PROCESO DE SOFTWARE 10
1.3.1 Modelo Lineal Secuencial 10
1.3.2 Modelo en Espiral 11
1.3.3 Modelo de Prototipos 12
1.3.4 El Modelo Incremental 13
1.4 BASE DE DATOS 13
1.4.2 Introducción y Definición de Base de Datos 13
1.4.3 DBMS (Database Management System o Sistema Manejador de BD) 14
1.4.4 Usuarios de la Base de Datos 15
1.5 MODELO DE DATOS 16
1.5.1 Modelo de Datos 16
1.5.2 Modelo Entidad-Relación (E-R) 16
1.5.3 Modelo Relacional 17
1.5.4 Llave Primarias y Foráneas 17
1.6 NORMALIZACIÓN 18
1.6.1 Definición de normalización 18
1.7 ARQUITECTURA CLIENTE-SERVIDOR 19
1.7.1 Código Abierto (Open Source) 19
1.7.2 Apache 19
1.7.3 PHP 19
1.7.4 Mysql 21

CAPITULO 2: ANÁLISIS DE REQUERIMIENTOS 22
2.1 INTRODUCCIÓN 23
2.2 PLANTEAMIENTO DEL PROBLEMA 23
2.3 OBJETIVO GENERAL 24
2.4 REQUERIMIENTOS DEL SISTEMA 25
2.4.1 Requerimientos Funcionales 25
2.4.2 Requerimientos No Funcionales 26
2.5 ALCANCE 27
2.6 METODOLOGÍA 27
2.7 DISEÑO DEL SISTEMA 28
2.7.1 Casos de Uso 28

CAPITULO 3: DISEÑO DE LA BASE DE DATOS 34
3.1 DISEÑO CONCEPTUAL DE LA BASE DE DATOS 35
3.1.1 Modelo Entidad-Relación 35
3.1.2 Descripción de Entidades y Relaciones 36
3.1.3 Diseño Lógico 36
3.1.4 Normalización 38

3.2 DISEÑO DE LA APLICACIÓN 39
3.2.1 Diagramas de Estado, Secuencia y Colaboración 39
3.2.2 Diagrama de Clases 48

CAPITULO 4: IMPLEMENTACIÓN Y PRUEBAS 48
4.1 IMPLEMENTACIÓN 49
4.2 IMPLEMENTACIÓN DE LA INTERFAZ 50
4.2.1 Página de Inicio 50
4.2.1 Interfaz del Usuario 53
4.2.2 Interfaz del Administrador 56

CONCLUSIONES 59

TRABAJOS A FUTURO 59

APÉNDICE A 60

BIBLIOGRAFIA 66

INTRODUCCION

La adecuada elaboración de un diseño de bases de datos siempre ha sido importante. En la actualidad poseer, administrar y transmitir formación, es de tal importancia ya que toda la humanidad se ve y seguirá viendo afectada, influida y posiblemente dominada por quienes tienen, administran y transmiten este recurso, razón por lo cual a esta época se le denomina “Sociedad de la Información“ o “Revolución Electrónica”, éste último debido a la facilidad con que se procesa y transmite la información por medio de los sistemas modernos basados en dispositivos electrónicos.

Uno de los aspectos más abstractos e importantes de la información es que su valor puede decrecer o incluso desaparecer un tiempo después, por tanto, la información debe ser rápida, segura, veraz y oportuna, de lo contrario no es de utilidad; en consecuencia, la necesidad de disponer rápidamente de información para la solución de problemas de modo inmediato, debe basarse en el gran desarrollo científico y tecnológico, donde los sistemas de información son un conjunto de métodos, herramientas y datos que están diseñados para actuar coordinada y lógicamente, para capturar, almacenar, analizar, transformar y presentar la información con el fin de satisfacer múltiples propósitos.

Con lo anteriormente expuesto, en la actualidad es sorprendente ver la cantidad de sistemas de bases de datos que se encuentran funcionando sin haber pasado por un proceso adecuado de diseño, sobre todo en sistemas pequeños, donde el diseñador y el programador son la misma persona, por lo que se inicia la programación teniendo un análisis de requerimientos mínimos y en consecuencia, no teniendo la planeación y documentación para llevar a cabo adecuadamente la fase de diseño. Adicionalmente, bajo la prioridad del cliente de obtener del software final en el menor tiempo posible, el proceso de diseño pasa a ser más un proceso mental soportado en algunas notas y no en documentación apegada a metodologías ampliamente probadas. Bajo estas circunstancias, para obtener la propuesta final, en la mayoría de las ocasiones se basan en reiteradas observaciones que los usuarios van haciendo en cada evaluación, lo cual lo vuelve un proceso largo y tedioso.

El presente trabajo centra la importancia del diseño conceptual de bases de datos desde el enfoque del modelo entidad-relación y su influencia, ya que es primordial la adecuada definición de las estructuras a utilizar a fin de obtener un sistema robusto y eficiente, con base en metodologías ampliamente probadas.

Hoy en día, toda persona hace uso de diferentes equipos de cómputo y los sitios de su interés los anota en una hoja o en su agenda, en muchas ocasiones se pierde la información y debe perder tiempo buscando aquellas direcciones que le interesaron, es por eso que se requiere de un sitio donde se pueda guardar dicha información, pero además que pueda compartir ligas cuando sea necesario.

El objetivo es desarrollar un sistema de información basado en Web donde se puedan mantener los marcadores sociales y favoritos para los alumnos y académicos de la Facultad de Ciencias de la Computación, para que se tenga siempre la información de cada usuario de sus sitios de interés no importando que navegador, sistema operativo o lugar donde se encuentre y pueda acceder a dichos sitios.

Así con el objetivo de tener un sitio dentro de la Facultad donde alumnos y académicos puedan guardar aquellos sitios que sean de su interés y con la conexión a Internet accedan a dichos sitios y es como surge el planteamiento de generar un Sistema Web que mantenga los sitios de interés a un usuario, a un grupo de estudiantes o académicos que comparta, incluso que toda la comunidad vea, para lo cual se debe dar la posibilidad de que dichos sitios sean visitados por el usuario, un grupo de usuarios o bien al público en general y que los usuarios tengan la posibilidad de lograr mantener información relevante para ellos.

CAPITULO 1

MARCO TEÓRICO

1.1 INGENIERÍA DEL SOFTWARE

1.1.1 Definición Del Término Ingeniería De Software

De acuerdo a Pressman [1] La Ingeniería del Software es la tecnología que comprende un “Proceso”, un juego de métodos y un conjunto de herramientas.

La Ingeniería del Software surge de la Ingeniería de Sistemas y de Hardware, abarca un conjunto de tres elementos clave: métodos, herramientas y procedimientos, que facilitan al gestor controlar el proceso del desarrollo software de alta calidad de una forma productiva.

Los *métodos* de la Ingeniería del Software indican "cómo" construir técnicamente el software. Los métodos abarcan un amplio espectro de tareas que incluyen: planificación y estimación de proyectos, análisis de los requisitos del sistema y del software, diseño de estructuras de datos, arquitectura de programas y procedimientos algorítmicos, codificación, prueba y mantenimiento.

Las *herramientas* de la Ingeniería del Software suministran un soporte automático o semiautomático para los métodos. Existen herramientas para soportar cada uno de los métodos mencionados anteriormente. Cuando se integran las herramientas de forma que la información creada por una herramienta pueda ser usada por otra, se establece un sistema para el soporte del desarrollo del software, llamado Ingeniería del Software Asistida por Computadora (del inglés, CASE: Computer Aided Software Engineering) [1].

La ingeniería del software aparece como consecuencia de un proceso denominado: ingeniería de sistemas. La ingeniería de sistemas ayuda a traducir las necesidades del cliente ya que comprende varios componentes: Hardware, Software, Personas, Bases de Datos, Documentación y Procedimientos.

Así se identifican las necesidades del cliente y se determina la viabilidad económica y técnica para asignar funciones y rendimientos al software, hardware, personas y bases de datos.

La ingeniería del sistema demanda una intensa comunicación entre el cliente y el ingeniero del sistema y esto se realiza a través de la ingeniería de requisitos que comprende: identificación, análisis y negociación, especificación, modelización, validación y gestión.

Los procedimientos son la aplicación de los métodos utilizando las herramientas, es decir, son los elementos de unión entre los métodos y las herramientas. Definen la secuencia en la que se aplican los métodos, ayudan a asegurar la calidad y coordinar los cambios. La ingeniería de software sigue evolucionando, el desarrollo de software orientado a objetos comenzó en los

años ochenta como una etapa natural de los métodos estructurados y actualmente UML (Unified Modeling Language) ha emergido como una unificación de los diversos métodos orientados a objetos y se está convirtiendo en un estándar.

1.2 RUP (Rational Unified Process o Proceso Unificado de Rational) Y UML (Unified Modeling Language o Lenguaje Unificado de Modelado)

1.2.1 Proceso Unificado de Rational

El Proceso Unificado de Rational (RUP, Rational Unified Process) es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. RUP está basado en el seguimiento de una serie de normas o “mejores prácticas” aplicadas a cuatro etapas del desarrollo software: iniciación, elaboración, construcción y transición.

1.2.2 Características Esenciales

El proceso de software propuesto por RUP tiene tres características esenciales: está dirigido por los Casos de Uso, está centrado en la arquitectura, y es iterativo e incremental.

RUP identifica 6 best practices con las que define una forma efectiva de trabajar para los equipos de desarrollo de software.

- Gestión de requisitos: RUP brinda una guía para encontrar, organizar, documentar, y seguir los cambios de los requisitos funcionales y restricciones. Utiliza una notación de Caso de Uso y escenarios para representar los requisitos.

- Desarrollo de software iterativo: Desarrollo del producto mediante iteraciones con hitos bien definidos, en las cuales se repiten las actividades pero con distinto énfasis, según la fase del proyecto.

- Desarrollo basado en componentes: Esta característica en un proceso de desarrollo permite que el sistema se vaya creando a medida que se obtienen o se desarrollan sus componentes.

- Modelado visual (usando UML): UML es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema

software. El modelado visual ayuda a mejorar la capacidad del equipo para gestionar la complejidad del software.

- Verificación continúa de la calidad: Es importante que la calidad de todos los artefactos se evalúe en varios puntos durante el proceso de desarrollo, especialmente al final de cada iteración.

- Gestión de los cambios: Los artefactos software cambian no sólo debido a acciones de mantenimiento posteriores a la entrega del producto, sino que durante el proceso de desarrollo, especialmente importantes por su posible impacto son los cambios en los requisitos.

1.2.3 Ciclo de Vida

El ciclo de vida RUP es una implementación del Desarrollo en espiral. Fue creado ensamblando los elementos en secuencias semi-ordenadas. El ciclo de vida organiza las tareas en fases e iteraciones.

El RUP divide el proceso de desarrollo en ciclos, teniendo un producto final al final de cada ciclo, cada ciclo se divide en fases que finalizan con un hito donde se debe tomar una decisión importante:

- Concepción: se hace un plan de fases, se identifican los principales casos de uso y se identifican los riesgos

- Elaboración: se hace un plan de proyecto, se completan los casos de uso y se eliminan los riesgos

- Construcción: se concentra en la elaboración de un producto totalmente operativo y eficiente y el manual de usuario

- Implementación: se Instala el producto en el cliente y se entrena a los usuarios. Como consecuencia de esto suelen surgir nuevos requisitos a ser analizados.

La duración y esfuerzo dedicado en cada fase es variable dependiendo de las características del proyecto.

El Proceso Unificado se enfoca en la arquitectura como el centro del desarrollo para asegurar que el desarrollo basado en componentes sea clave para un alto nivel de reuso. Se considera que hay cuatro perspectivas de arquitectura que cumplen los requerimientos de una empresa:

- Arquitectura de Negocios: Desarrolla una imagen clara de los procesos de flujo de trabajo de la organización y de cómo son apoyados por una infraestructura tecnológica basada en servicios.

- Arquitectura de Aplicación: Adopta un modelo de aplicación de toda la empresa para diseñar y desarrollar sistemas de negocios que puedan compartir un conjunto de componentes back-end de alto valor.

- Arquitectura de Información: Define qué información es necesaria para apoyar el proceso de negocios y como poner esa información eficientemente en manos de quienes que la necesitan sin crear islas de datos inaccesibles ni sistemas redundantes.
- Arquitectura Tecnológica: Define los estándares y guías para la adquisición y despliegue de herramientas, bloques de construcción de aplicaciones, servicios de infraestructura, componentes de conectividad de red y plataformas cliente servidor.

El Proceso Unificado es un proceso porque "define quién está haciendo qué, cuándo lo hacer y cómo alcanzar cierto objetivo, en este caso el desarrollo de software".

1.2.4 Que es UML

UML (Unified Modeling Language) es un lenguaje que permite modelar, construir y documentar los elementos que forman un sistema software orientado a objetos. Se ha convertido en el estándar de facto de la industria, debido a que ha sido concebido por los autores de los tres métodos más usados de orientación a objetos: Grady Booch, Ivar Jacobson y Jim Rumbaugh. Estos autores fueron contratados por la empresa Rational Software Co. para crear una notación unificada en la que basar la construcción de sus herramientas CASE.

Esta notación ha sido ampliamente aceptada debido al prestigio de sus creadores y debido a que incorpora las principales ventajas de cada uno de los métodos particulares en los que se basa: Booch, OMT y OOSE. UML ha puesto fin a las llamadas "guerras de métodos" que se han mantenido a lo largo de los 90, en las que los principales métodos sacaban nuevas versiones que incorporaban las técnicas de los demás. Con UML se fusiona la notación de estas técnicas para formar una herramienta compartida entre todos los ingenieros software que trabajan en el desarrollo orientado a objetos [3].

El objetivo principal cuando se empezó a gestar UML era posibilitar el intercambio de modelos entre las distintas herramientas CASE orientadas a objetos del mercado. Para ello era necesario definir una notación y semántica común. En la Figura 1 se puede ver cuál ha sido la evolución de UML hasta la creación de UML 1.1.

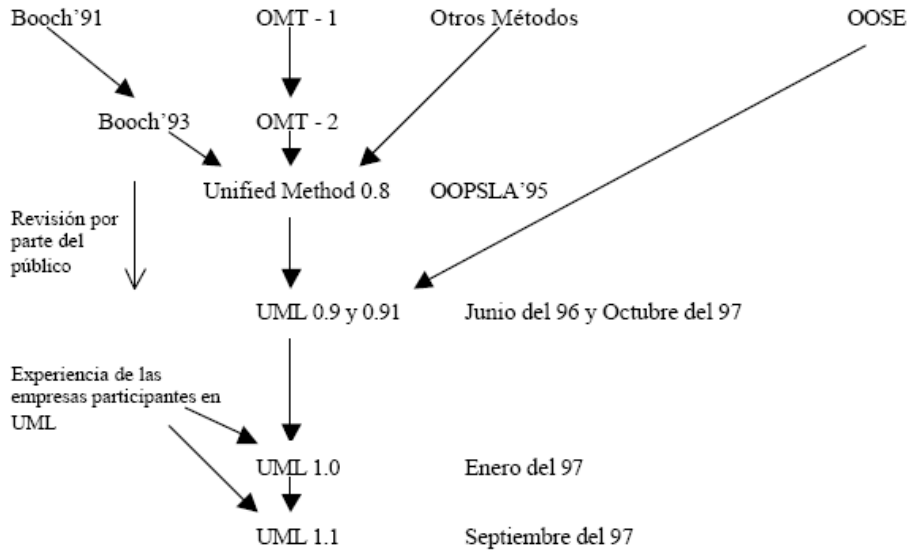

Figura 1. Historia de UML

Hay que tener en cuenta que el estándar UML no define un proceso de desarrollo específico, tan solo se trata de una notación [3].

1.2.5 Modelos de UML

Un modelo representa a un sistema software desde una perspectiva específica. Al igual que la planta y el alzado de una figura en dibujo técnico nos muestran la misma figura vista desde distintos ángulos, cada modelo nos permite fijarnos en un aspecto distinto del sistema. Los modelos de UML son los siguientes:

- Diagrama de Estructura Estática.
- Diagrama de Casos de Uso.
- Diagrama de Secuencia.
- Diagrama de Colaboración.
- Diagrama de Estados.

Diagramas de estructura estática

Con el nombre de Diagramas de Estructura Estática se engloba tanto al Modelo Conceptual de la fase de Análisis como al Diagrama de Clases de la fase de Diseño. Ambos son distintos conceptualmente, mientras el primero modela elementos del dominio el segundo presenta los elementos de la solución software. Sin embargo, ambos comparten la misma notación para los elementos que los forman (clases y objetos) y las relaciones que existen entre los mismos (asociaciones).

- Clases: Una clase se representa mediante una caja subdividida en tres partes: En la superior se muestra el nombre de la clase, en la media los atributos y en la inferior las operaciones.

- Objetos: Un objeto se representa de la misma forma que una clase. En el compartimiento superior aparece el nombre del objeto junto con el nombre de la clase subrayado.

- Asociaciones: Las asociaciones entre dos clases se representan mediante una línea que las une. Puede tener una serie de elementos gráficos que expresan características particulares de la asociación.

Diagrama de casos de uso

Un Diagrama de Casos de Uso muestra la relación entre los actores y los casos de uso del sistema. Representa la funcionalidad que ofrece el sistema en lo que se refiere a su interacción externa. En un Diagrama de Casos de Uso son: actores, casos de uso y relaciones entre casos de uso.

- Actores: Entidad externa al sistema que realiza algún tipo de interacción con el mismo. Se representa mediante una figura humana dibujada con palos.

- Casos de Uso: Descripción de la secuencia de interacciones que se producen entre un actor y el sistema, cuando el actor usa el sistema para llevar a cabo una tarea específica. Se representa mediante una elipse con el nombre del caso de uso en su interior.

- Relaciones entre Casos de Uso: Entre dos casos de uso puede haber las relaciones Extiende (cuando un caso de uso especializa a otro extendiendo su funcionalidad) o Usa (cuando un caso de uso utiliza a otro).

Diagrama de secuencia

Un diagrama de Secuencia muestra una interacción ordenada según la secuencia temporal de eventos. En particular, muestra los objetos participantes en la interacción y los mensajes que intercambian ordenados según su secuencia en el tiempo. El eje vertical representa el tiempo, y en el eje horizontal se colocan los objetos y actores participantes en la interacción, sin un orden prefijado. Cada objeto o actor tiene una línea vertical, y los mensajes se representan mediante flechas entre los distintos objetos. El tiempo fluye de arriba abajo. Se pueden colocar etiquetas (como restricciones de tiempo, descripciones de acciones, etc.) bien en el margen izquierdo o bien junto a las transiciones o activaciones a las que se refieren.

Diagrama de colaboración

Un Diagrama de Colaboración muestra una interacción organizada basándose en los objetos que toman parte en la interacción y los enlaces entre los mismos (en cuanto a la interacción se refiere). A diferencia de los Diagramas de Secuencia, los Diagramas de Colaboración muestran las relaciones entre los roles de los objetos. La secuencia de los mensajes y los

flujos de ejecución concurrentes deben determinarse explícitamente mediante números de secuencia.

Diagrama de estados

Un Diagrama de Estados muestra la secuencia de estados por los que pasa un caso de uso o un objeto a lo largo de su vida, indicando qué eventos hacen que se pase de un estado a otro y cuáles son las respuestas y acciones que genera. En cuanto a la representación, un diagrama de estados es un grafo cuyos nodos son estados y cuyos arcos dirigidos son transiciones etiquetadas con los nombres de los eventos.

Un diagrama de estados puede representar ciclos continuos o bien una vida finita, en la que hay un estado inicial de creación y un estado final de destrucción (del caso de uso o del objeto). El estado inicial se muestra como un círculo sólido y el estado final como un círculo sólido rodeado de otro círculo. En realidad, los estados inicial y final son pseudoestados, pues un objeto no puede “estar” en esos estados, pero nos sirven para saber cuáles son las transiciones inicial y final(es) [3].

1.3 MODELOS DE PROCESO DE SOFTWARE

1.3.1 Modelo Lineal Secuencial

Llamado algunas veces “Ciclo de vida básico“(Figura 2), este modelo secuencial sugiere un enfoque sistemático y secuencial para el desarrollo del software que comienza en un nivel de sistemas y progresa con el análisis, diseño, codificación, pruebas y mantenimiento.

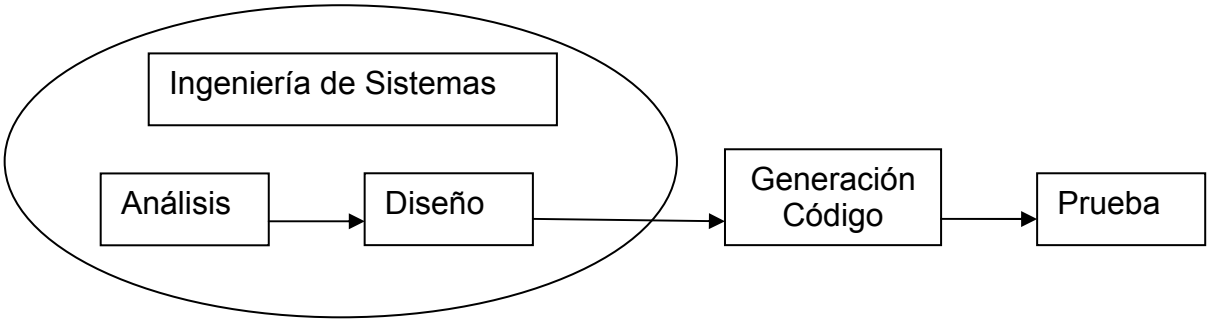

Figura 2. Modelo Secuencial

Ingeniería de sistemas: Consiste en establecer todos los requisitos de todos los elementos del sistema ya que el software interconectará con otros elementos como lo son el hardware, personas, bases de datos y procesos.

Análisis: Se analizan los requisitos del software, dominio de información, la función requerida, comportamiento, rendimiento e interconexiones.

Diseño: En esta etapa del diseño del software se centra la estructura de datos, arquitectura de software, representaciones de interfaz y detalle procedimental (algoritmos).

Generación de Código: Fase dependiente completamente de la fase de Diseño ya que si el diseño fue llevado a cabo de una forma detallada, la generación del código se realiza mecánicamente.

Pruebas: Consiste en probar el código generado y se centra en los procesos lógicos internos del software, asegurándose que las sentencias se han comprobado y básicamente asegurarse de que las entradas definidas produce los resultados reales de acuerdo con los resultados requeridos [4].

1.3.2 Modelo en Espiral

El Modelo en Espiral (Figura 3), es un modelo de proceso de software evolutivo que conjuga la naturaleza iterativa de construcción de prototipos con los aspectos controlados y sistemáticos del modelo lineal secuencial.

Las prioridades de este método iniciando de dentro hacia fuera son:

1.- Proyecto de desarrollo de conceptos.
2.- Proyecto de desarrollo de nuevos productos.
3.-Proyecto de mejora de productos.
4.- Proyecto de mantenimiento de producto.

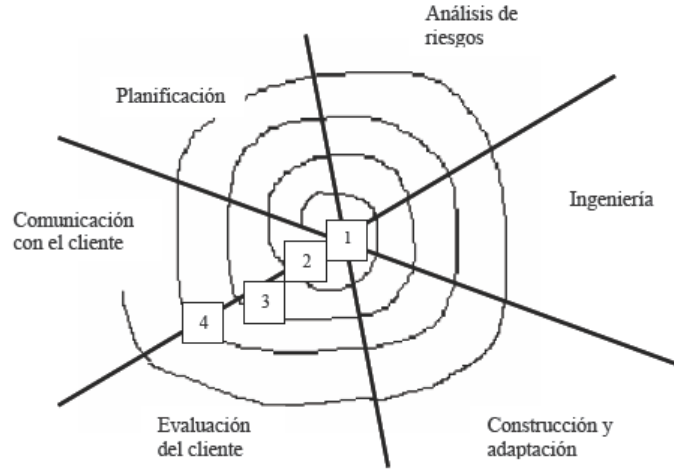

Figura 3. Modelo en Espiral

En el modelo espiral, el software se desarrolla en una serie de versiones incrementales. Durante las primeras versiones la versión puede ser un prototipo o bien un modelo en papel y durante las últimas iteraciones las versiones suelen ser cada vez más completas, hasta tener el sistema diseñado.

Este modelo se divide en un número de actividades que suelen ser 6 regiones de tareas:

- Comunicación con el cliente: las tareas requeridas para establecer comunicación entre el desarrollador y el cliente.
- Planificación: las tareas requeridas para definir recursos, el tiempo y otra información relacionadas con el proyecto.
- Análisis de riesgos: Las tareas requeridas para evaluar riesgos técnicos y de gestión, análisis de alternativas e identificación/resolución de riesgos.
- Ingeniería: las tareas requeridas para construir una o más representaciones de la aplicación.
- Construcción y acción: Las tareas requeridas para construir, probar, instalar y proporcionar soporte al usuario (por ejemplo: documentación y práctica)
- Evaluación del cliente: las tareas requeridas para obtener la reacción del cliente según la evaluación de las representaciones del software creadas durante la etapa de ingeniería e implementada durante la etapa de instalación [4].

1.3.3 Modelo de Prototipos

Este modelo de construcción de prototipos (Figura 4), inicia con la recolección de requisitos, el cliente y el ingeniero de sistemas definen los objetivos del sistema de ahí nace un diseño rápido de la construcción de un prototipo. Posteriormente lo evalúa el cliente/operador.

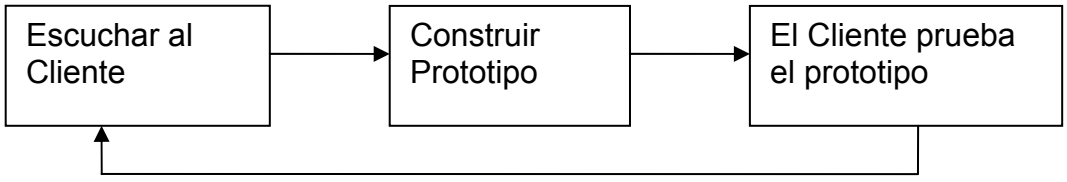

Figura 4. Modelo De Construcción De Prototipos

Si se trata de estos prototipos desechables generalmente, apenas pueden utilizarse, son demasiado lentos, demasiados grandes o muy poco operativos así que el destino de estos prototipos es tirarlos y volver a empezar. Sin embargo explicar esto a los clientes es complicado ya que el cliente pensará que con solo algunos ajustes del prototipo podría convertirse en el producto final [4].

1.3.4 El Modelo Incremental

El modelo incremental combina elementos del modelo lineal secuencial (aplicados repetidamente) con la filosofía interactiva de construcción de prototipos. El modelo incremental entrega el software en partes pequeñas, pero utilizables, llamadas “incrementos”.

La diferencia entre el modelo de prototipos y el modelo del proceso incremental se da en que el modelo incremental se centra en la entrega de un producto operacional con cada incremento, los primeros incrementos son versiones incompletas del producto final, pero proporcionan al usuario la funcionalidad que precisa y también una plataforma para la evaluación [4].

1.4 BASE DE DATOS

1.4.2 Introducción y Definición de Base de Datos

En el entorno informático, la gestión de bases de datos ha evolucionado desde ser una aplicación más disponible para las computadoras, a ocupar un lugar fundamental en los sistemas de información. En la actualidad, un sistema de información será más valioso cuanto de mayor calidad sea la base de datos que lo soporta, la cual resulta a su vez un componente fundamental del mismo, de tal forma que puede llegarse a afirmar que es imposible la existencia de un sistema de información sin una base de datos, que cumple la función de "memoria", en todas sus acepciones posibles, del sistema.

Las bases de datos almacenan, como su nombre dice, datos. Estos datos son representaciones de sucesos y objetos, a diferente nivel, existentes en el mundo real: en su conjunto, representan algún tipo de entidad existente. En el mundo real se tiene percepción sobre las entidades u objetos y sobre los atributos de esos objetos; en el mundo de los datos, hay registros de eventos y datos de eventos. Además, en ambos escenarios se puede incluso distinguir una tercera faceta: aquella que comprende las definiciones de las entidades externas, o bien las definiciones de los registros y de los datos.

La transferencia entre las entidades del mundo real, y sus características, y los registros contenidos en una base de datos, correspondientes a esas entidades, se alcanza tras un proceso lógico de abstracción, conjunto de tareas que suelen englobarse bajo el título de diseño de bases de datos. Sin embargo, es necesario definir, en primer lugar, qué es una base de datos, independientemente de su diseño y/o su orientación. Entre las numerosas definiciones que pueden encontrarse, pueden escogerse, por su exhaustividad, las siguientes:

- "Colección de datos correspondientes a las diferentes perspectivas de un sistema de información (de una empresa o institución), existentes en algún soporte de tipo físico (normalmente de acceso directo), agrupados en una organización integrada y centralizada en la que figuran no sólo los datos en sí, sino también las relaciones existentes entre ellos, y de forma que se minimiza la redundancia y se maximiza la independencia de los datos de las aplicaciones que los requieren" [5].

- "Una base de datos es una colección de datos estructurados según un modelo que refleje las relaciones y restricciones existentes en el mundo real. Los datos, que han de ser compartidos por diferentes usuarios y aplicaciones, deben mantenerse independientes de éstas, y su definición y descripción han de ser únicas estando almacenadas junto a los mismos. Por último, los tratamientos que sufran estos datos tendrán que conservar la integridad y seguridad de éstos" [6].

La segunda definición añade los objetivos que debe cumplir un sistema de gestión de bases de datos. Basta considerar que deben cumplir los objetivos de independencia de los datos (las aplicaciones no deben verse afectadas por cambios en la estructura de los datos), integridad de los datos (los datos deben cumplir ciertas restricciones que aseguren la correcta introducción, modificación y borrado de los mismos) y seguridad (establecer diferentes niveles de acceso a los datos a diferentes tipos de usuarios).

1.4.3 DBMS (Database Management System o Sistema Manejador de BD)

El DBMS (sistema de administración de bases de datos) es el software que maneja todo acceso a la base de datos.

1. Un usuario emite una petición de acceso, utilizando algún sublenguaje de datos especifico.

2. El DBMS intercepta esa petición y la analiza.

3. El DBMS inspecciona, en su momento, el esquema externo para el usuario, la transformación externa/conceptual correspondiente, el esquema conceptual, la transformación conceptual/interna y la definición de la estructura de almacenamiento.

4. El DBMS ejecuta las operaciones necesarias sobre la base de datos almacenada.

El DBMS debe ser capaz de manejar peticiones para recuperar, actualizar o eliminar datos existentes en la base de datos o agregar nuevos datos a esta.

Optimización y ejecución.

Las peticiones DML (Lenguaje de Manipulación de Datos), planeadas o no planeadas, deber ser procesadas por el componente optimizador, cuya finalidad es determinar una forma eficiente de implementar la petición.

Seguridad e integridad de datos.

El DBMS debe vigilar las peticiones del usuario y rechazar todo intento de violar las restricciones de seguridad y de integridad definidas por el administrador de base de datos (DBA, Data Base Administrator). Estas tareas pueden realizarse durante el tiempo de compilación, de ejecución o entre ambos.

Recuperación de datos y concurrencia.

Debe incorporar ciertos controles de recuperación y Concurrencia.

Diccionario de datos.

El DBMS debe proporcionar una función de diccionario de datos. Este diccionario puede verse como una base de datos (como una base de datos del sistema y no del usuario). Contiene “datos acerca de los datos” (metadatos o descriptores; es decir. Definiciones de otros objetos del sistema, en lugar de simples “datos en bruto”).

Rendimiento.

El DBMS debe realizar todas las tareas antes identificadas de la manera más eficiente posible.

1.4.4 Usuarios de la Base de Datos

En consonancia con las posibles, y diferentes, vistas externas, se pueden identificar varios tipos de usuarios.

En primer lugar, los usuarios finales, que hacen un uso limitado de las capacidades del sistema, normalmente referentes a introducción, manipulación y consulta de los datos. Los usuarios finales pueden ser sofisticados o especializados e ingenuos, dependiendo de su nivel de interacción con el sistema.

En segundo lugar hay que citar a los programadores de base de datos, encargados de escribir aplicaciones limitadas, mediante el lenguaje de programación facilitado por el DBMS, normalmente algún el lenguaje de cuarta generación, que faciliten la ejecución de tareas por parte de los usuarios finales.

Por último, el DBA cumple las importantes funciones de crear y almacenar las estructuras de la base de datos, definir las estrategias de respaldo y recuperación, vincularse con los usuarios y responder a sus cambios de

requerimientos, y definir los controles de autorización y los procedimientos de validación.

1.5 MODELO DE DATOS

1.5.1 Modelo de Datos

En el proceso de abstracción que conduce a la creación de una base de datos desempeña una función prioritaria el modelo de datos. El modelo de datos es el enfoque utilizado para la representación de las entidades y sus características dentro de la base de datos.

El punto clave en la construcción de la base de datos será el modelo de datos. Se denomina modelo:

"...al instrumento que se aplica a una parcela del mundo real (universo del discurso) para obtener una estructura de datos a la que denominamos esquema. Esta distinción entre el modelo (instrumento) y el esquema (resultado de aplicar el instrumento) es importante... Es importante también distinguir entre mundo real y universo del discurso, ya que este último es la visión que del mundo real tiene el diseñador... podemos definir un modelo de datos como un conjunto de conceptos, reglas y convenciones que nos permiten describir los datos del universo del discurso."

Los objetivos del modelo de datos son dos:

- Formalización: definir formalmente las estructuras permitidas y las restricciones a fin de representar los datos de un SI.

- Diseño: el modelo resultante es un elemento básico para el desarrollo de la metodología de diseño de la base de datos.

1.5.2 Modelo Entidad-Relación (E-R)

El modelo de datos Entidad - Relación (E -R) está basado en una percepción del mundo real que consta de una colección de objetos básicos, llamados entidades, y de relaciones entre estos objetos. Las entidades se describen en una base de datos mediante un conjunto de atributos [7].

Una relación es una asociación entre varias entidades. Por ejemplo, una relación impositor asocia un cliente con cada cuenta que tiene. El conjunto de todas las entidades del mismo tipo, y el conjunto de todas las relaciones del

mismo tipo, se denominan respectivamente conjunto de entidades y conjunto de relaciones [8].

La estructura lógica general de una base de datos se puede expresar gráficamente mediante un diagrama E - R, que consta de los siguientes componentes:

- Rectángulos, que representan conjuntos de entidades.
- Elipses, que representan atributos.
- Rombos, que representan relaciones entre conjuntos de entidades.
- Líneas, que unen los atributos con los conjuntos de entidades y los conjuntos de entidades con las relaciones.

Cada componente se etiqueta con la entidad o relación que representa.

1.5.3 Modelo Relacional

En el modelo relacional se utiliza un grupo de tablas para representar los datos y las relaciones entre ellos. El modelo relacional es un ejemplo de un modelo basado en registros. Los modelos basados en registros se denominan así porque la base de datos se estructura en registros de formato fijo de varios tipos [8].

No es difícil ver cómo se pueden almacenar las tablas en archivos. Por ejemplo un carácter especial (como una coma) se puede usar para delimitar los diferentes atributos de un registro, y otro carácter especial (como un carácter de nueva línea) se puede usar para delimitar registros. El modelo de datos relacional es el modelo de datos más ampliamente usado, y una amplia mayoría de sistemas de bases de datos actuales se basan en el modelo relacional.

1.5.4 Llave Primarias y Foráneas

Llave primaria

Es una columna o grupo de columnas que identifican de manera única a cada renglón de una tabla. Cada tabla debe tener una llave primaria y una llave primaria debe ser única. Una tabla puede tener más de una columna o combinación de columnas que pueden servir como llave primaria de la tabla, cada una de estas es llamada llave candidata o alterna.

Llaves foráneas

Una llave foránea es una columna o combinación de columnas en una tabla, que se refieren a una llave primaria en la misma o en la otra tabla. Una llave foránea debe coincidir con un valor de una llave primaria existente.

1.6 NORMALIZACIÓN

1.6.1 Definición de normalización

Normalización es un conjunto de reglas que sirven para ayudar a los diseñadores a desarrollar un esquema que minimice los problemas de lógica. Cada regla está basada en la que le antecede. El proceso de normalización tiene un nombre y una serie de reglas para cada fase.

Una ventaja de la normalización de la base de datos es el consumo de espacio. Una base de datos normalizada puede ocupar menos espacio en disco que una no normalizada. Hay menos repetición de datos, lo que tiene como consecuencia un mucho menor uso de espacio en disco.

Existen básicamente tres niveles de normalización: Primera Forma Normal (1NF), Segunda Forma Normal (2NF) y Tercera Forma Normal (3NF). Cada una de estas formas tiene sus propias reglas. Cuando una base de datos se conforma a un nivel, se considera normalizada a esa forma de normalización.

- Primera Forma Normal: Establece que las columnas repetidas deben eliminarse y colocarse en tablas separadas.

- Segunda Forma Normal: Establece que todas las dependencias parciales se deben eliminar y separar dentro de sus propias tablas. Una dependencia parcial es un término que describe a aquellos datos que no dependen de la clave de la tabla para identificarlos.

- Tercera Forma Normal: Establece que hay que eliminar y separar cualquier dato que no sea clave. El valor de esta columna debe depender de la clave. Todos los valores deben identificarse únicamente por la clave.

Existen seis niveles de normalización: Forma Normal Boyce-Codd, Cuarta Forma Normal (4NF), Quinta Forma Normal (5NF) o Forma Normal de Proyección-Unión, Forma Normal de Proyección-Unión Fuerte, Forma Normal de Proyección-Unión Extra Fuerte y Forma Normal de Clave de Dominio. Estas formas de normalización pueden llevar las cosas más allá de lo que necesita. Éstas existen para hacer una base de datos realmente relacional. Tienen que ver principalmente con dependencias múltiples y claves relacionales.

Las primeras tres formas proveen suficiente nivel de normalización para cumplir con las necesidades de la mayoría de las bases de datos [9].

1.7 ARQUITECTURA CLIENTE-SERVIDOR

1.7.1 Código Abierto (Open Source)

Código abierto (del inglés open source) es el término con el que se conoce al software distribuido y desarrollado libremente. Fue utilizado por primera vez en 1998 por algunos usuarios de la comunidad del software libre, tratando de usarlo como reemplazo al ambiguo nombre original en inglés del software libre (free software).Free en inglés puede significar diferentes cosas: gratuidad y libertad. Por ello, por un lado, permite pensar en "software por el que no hay que pagar" (software gratuito) y, por otro, se adapta al significado que se pretendió originalmente (software que posee ciertas libertades).

En la actualidad open source es utilizado para definir un movimiento nuevo de software, diferente al movimiento del Software Libre, incompatible con este último desde el punto de vista filosófico, y completamente equivalente desde el punto de vista práctico, de hecho, ambos movimientos trabajan juntos en el desarrollo práctico de proyectos.

La idea detrás del open source es bien sencilla: cuando los programadores en Internet pueden leer, modificar y redistribuir el código fuente de un programa, éste evoluciona, se desarrolla y mejora. Los usuarios lo adaptan a sus necesidades, corrigen sus errores a una velocidad impresionante, mayor a la aplicada en el desarrollo de software convencional o cerrado, dando como resultado la producción de un mejor software [10].

1.7.2 Apache

El servidor HTTP Apache es un software (libre) servidor HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual.

Apache presenta entre otras características mensajes de error altamente configurables, bases de datos de autenticación y negociado de contenido. Apache tiene amplia aceptación en la red: en el 2005, Apache es el servidor HTTP más usado, siendo el servidor HTTP del 48% de los sitios Web en el mundo y decreciendo su cuota de mercado [11].

1.7.3 PHP

PHP es el heredero de un producto anterior, llamado PHP/FI. PHP/FI fue creado por Rasmus Lerdorf en 1995, inicialmente como un simple conjunto de scripts de PERL para controlar los accesos a su trabajo online.

PHP permite embeber pequeños fragmentos de código dentro de la Página HTML y realizar determinadas acciones de una forma fácil y eficaz sin tener que generar programas programados íntegramente en un lenguaje distinto al HTML. Por otra parte, y es aquí donde reside su mayor interés con respecto a los lenguajes pensados para los CGI, PHP ofrece un sinfín de funciones para la explotación de bases de datos de una manera llana, sin complicaciones.

Como funcionalidades primordiales en PHP se consideran:

- Funciones de correo electrónico.
- Gestión de bases de datos.
- Gestión de archivos.
- Tratamiento de imágenes.

Los principales usos del PHP son los siguientes:

- Programación de páginas Web dinámicas, habitualmente en combinación con el motor de base datos MySQL.
- Programación en consola, al estilo de Perl o Shell scripting.
- Creación de aplicaciones gráficas independientes del navegador, por medio de la combinación de PHP y GTK (GIMP Tool Kit).

Ventajas de PHP

- Es un lenguaje multiplataforma.
- Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL.
- Leer y manipular datos desde diversas fuentes, incluyendo datos que pueden ingresar los usuarios desde formularios HTML.
- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (Llamados ext's o extensiones).
- Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.

Es libre, por lo que se presenta como una alternativa de fácil acceso para todos. Permite las técnicas de Programación Orientada a Objetos, crear los formularios para la Web, Biblioteca nativa de funciones sumamente amplia e incluida, no requiere definición de tipos de variables ni manejo detallado del bajo nivel [12].

1.7.4 Mysql

MySQL es un gestor de base de datos sencillo de usar y increíblemente rápido. También es uno de los motores de base de datos más usados en Internet, la principal razón de esto es que es gratis para aplicaciones no comerciales.

Las características principales de MySQL son:

- Gestor de base de datos: Una base de datos es un conjunto de datos y un gestor de base de datos es una aplicación capaz de manejar este conjunto de datos de manera eficiente y cómoda.

- Base de datos relacional: Una base de datos relacional es un conjunto de datos que están almacenados en tablas entre las cuales se establecen unas relaciones para manejar los datos de una forma eficiente y segura. Para usar y gestionar una base de datos relacional se usa el lenguaje estándar de programación SQL.

- Open Source: El código fuente de MySQL se puede descargar y está accesible a cualquiera, por otra parte, usa la licencia GPL para aplicaciones no comerciales.

- Base de datos muy rápida, segura y fácil de usar: Gracias a la colaboración de muchos usuarios, la base de datos se ha ido mejorando optimizándose en velocidad. Por eso es una de las bases de datos más usadas en Internet [13].

CAPITULO 2

ANÁLISIS DE REQUERIMIENTOS

2.1 INTRODUCCIÓN

El diseño de una base de datos empieza con las entrevistas al usuario y las observaciones pertinentes, examinando las formas necesarias, preguntando a los usuarios finales, analizando la organización y el flujo de la información, es posible capturar los requisitos de los usuarios y dividirlos en contextos más o menos independientes. La abstracción de esas porciones limitadas del sistema de información permite una representación formal de los objetos, sus propiedades y sus interrelaciones, y las actividades que involucran esos objetos.

Así se puede realizar una representación de cada porción del modelo, obteniendo un esquema externo con aspectos estáticos y dinámicos. La selección apropiada de los detalles a modelar depende de las transacciones que el usuario tiene en mente, y más generalmente del propósito de la base de datos. Como salida, la primera fase del diseño de base de datos debe dar un conjunto de vistas externas para las características estáticas y los aspectos dinámicos.

La siguiente fase es integrar las diferentes vistas de la base de datos. La unión de las vistas no es suficiente, porque conceptos similares pueden describirse en diferentes vistas. Es muy necesario entender el significado de cada objeto y atributo, y aislar los objetos y los atributos que tienen significados similares o diferentes.

La fase de integración termina con la elaboración de un primer esquema global de la base de datos. Esta fase inicia con el previo esquema conceptual, ejecuta una primera transformación en las relaciones, y entonces se aplica la teoría de dependencias para mejorar el esquema.

El propósito es reducir la redundancia de los datos y evitar anomalías de actualización. Aunque teóricamente es un método bien fundamentado, el proceso de normalización es tedioso para aplicarlo, así que muchos diseñadores lo evitan.

La última fase es para especificar la representación física de la base de datos. La base para la optimización es el conocimiento de las transacciones más frecuentes. Ahora bien, las fases de captura y abstracción de los requisitos del usuario, así como la integración de las diferentes vistas externas corresponden a lo que es propiamente el análisis del sistema.

2.2 PLANTEAMIENTO DEL PROBLEMA

Hoy en día, toda persona hace uso de diferentes equipos de cómputo y los sitios que sean de sus interés se anotan en una hoja o en su agenda, en

muchas ocasiones se puede perder la información y debe perder tiempo buscando aquellas direcciones (ligas) que son de su interés, es por eso que se requiere de un sitio donde se pueda guardar dicha información, pero además que pueda compartir la direcciones (ligas) cuando sea necesario.

Así con el objetivo de tener un sitio dentro de la Facultad donde alumnos y académicos puedan guardar aquellos sitios que sean de su interés y con la conexión a Internet accedan a dichos sitios y es como surge el planteamiento de generar un Sistema Web que mantenga los sitios de interés a un usuario, a un grupo de estudiantes o académicos que comparta, incluso que toda la comunidad vea, para lo cual se debe dar la posibilidad de que dichos sitios sean visitados por el usuario, un grupo de usuarios o bien al público en general y que los usuarios tengan la posibilidad de lograr mantener información relevante para ellos.

2.3 OBJETIVO GENERAL

Desarrollar un sistema de información basado en Web donde se puedan mantener los marcadores sociales y favoritos para los alumnos y académicos de la Facultad de Ciencias de la Computación, para que se tenga siempre la información de cada usuario de sus sitios de interés no importando que navegador, sistema operativo o lugar donde se encuentre y pueda acceder a dichos sitios.
Objetivos Específicos:

- Elaborar la especificación de requerimientos del proyecto
- Realizar el análisis de los requerimientos usando UML
- Diseñar la base de datos.
- Diseñar la aplicación basada en Web usando UML.
- Implementar los diseños realizados usando programación orientada a objetos en PHP.
- Realizar pruebas del sistema.
- Escritura de la Documentación.

En términos generales, el objetivo es tener una página Web en la cual maestros y alumnos puedan ingresar y puedan realizar las siguientes operaciones: Guardar direcciones, Mostrar todas las direcciones del usuario, Actualizarlas, Borrarlas, y en caso de que el usuario no quiera seguir usando el sistema, dar de baja al usuario.

2.4 REQUERIMIENTOS DEL SISTEMA

2.4.1 Requerimientos Funcionales

Alta de usuario

- El sistema debe permitir que cualquier usuario pueda darse de alta, este debe proporcionar sus datos personales: nombre, apellidos, dirección de correo. Como también los datos que se van a utilizar en el sistema: su ID y su password, estos dos últimos para poder ingresar al sistema.

- El usuario debe de aceptar un acuerdo de condiciones de uso, en el cual explica el tipo de contenido que no esta permitido y en el cual el sitio no asume la responsabilidad de el contenido que tengan los usuarios.

Autentificación

- Se requerirá de la autentificación de cada usuario para que este pueda ingresar al sistema y así poder manipular los dominios a su criterio.

Manipulación de datos

- El sistema almacenaran los dominios de los usuarios, este último puede poner algún nombre de referencia para cada dominio, para su conveniencia.

- Una ves que el dominio este almacenado, este podrá ser modificado o eliminado dependiendo del criterio de cada usuario.

- Se otorgan atributos públicos y privados de los dominios para que estos se puedan compartir o mantener privados.

Baja del sistema

- El sistema debe permitir que cualquier usuario pueda darse de baja del sistema. Proporcionando su ID y su password, y aceptar una confirmación de baja del sistema.

- El sistema pide la ID y el password para que solo el usuario el cual pertenece la cuenta pueda darse de baja del sistema.

2.4.2 Requerimientos No Funcionales

Escalabilidad

- El diseño debe contemplar el uso óptimo de recursos tales como conexiones a la base de datos.

- Contemplar en el diseño la clara partición entre datos, recursos y aplicaciones para optimizar la escalabilidad del sistema.

Disponibilidad

- La disponibilidad del sistema debe ser continua con un nivel de servicio para los usuarios de 7 días X 24 horas.

- Debe contemplar requerimientos de confiabilidad y consistencia de los componentes. En caso de fallas de algún componente, no debe haber pérdida de información.

Seguridad

- La solución debe reflejar patrones de seguridad teniendo en cuenta sensibilidad de la información que maneja de acuerdo a las especificaciones funcionales dadas, normas y estándares de seguridad requeridas por el sistema. En este la seguridad se basa en el password que el usuario proporciona al darse de alta en el sistema, y este es responsabilidad del mismo.

Mantenibilidad

- Se debe estructurar el código de una manera consistente y predecible. Para objetos que son frecuentemente manejados en la lógica, implementar las respectivas interfaces que aseguren su fácil implementación en el sistema.

- Asegurar que el diseño de las interfaces contemplen el que las propiedades públicas y los parámetros de los métodos sean de un tipo común (estandarizados).

- El sistema debe ser construido e implantado de tal manera que un cambio en los parámetros obligue a la generación de una nueva versión del modulo.

Desempeño

- La aplicación debe ofrecer un buen desempeño del sistema ante una alta demanda acorde a los requerimientos funcionales y no funcionales de la solución.

2.5 ALCANCE

El Sistema Web para mantener sus marcadores sociales y favoritos contemplará las siguientes opciones:

- El sistema debe permitir que cualquier usuario pueda darse de alta en este. Proporcionando sus datos personales: nombre, apellidos, dirección de correo. Como también los datos que se van a utilizar en el sistema: su ID y su password, estos dos últimos para poder ingresar al sistema.

- El sistema debe requerir la autentificación de cada usuario para que este pueda ingresar al sistema y así poder manipular las direcciones a su criterio.

- El sistema debe almacenar las direcciones de los usuarios, este último puede poner algún nombre de referencia para cada dominio, para su conveniencia.

- El sistema debe permitir otorgarle atributos públicos y privados a las direcciones para que estas se puedan compartir o mantener privados.

- El sistema debe permitir que cualquier usuario pueda darse de baja del sistema. Proporcionando su ID y su password, y aceptar una confirmación de baja del sistema.

2.6 METODOLOGÍA

Se usará el ciclo de vida del software con una metodología orientada a objetos.

- En la primera etapa del proyecto se realizara el análisis y diseño orientado a objetos con UML, análisis y diseño de bases de datos con PHP y MYSQL, uso de tecnologías de vanguardia para el desarrollo del front-end.

- En la segunda etapa, se realizará el análisis y el diseño del proyecto.

- En la tercera etapa El cuarto se realizara la implementación utilizando PHP y Mysql como lenguaje de programación.

- En la cuarta etapa se realizaran las pruebas del sistema como la elaboración del documento.

- En conclusión, la metodología se encuentra centrada en conocimientos obtenidos en clase, en las prácticas y en un seguimiento continuo mediante asesoría personal, para el desarrollo del sistema.

2.7 DISEÑO DEL SISTEMA

2.7.1 Casos de Uso

Un caso de uso es una técnica para la captura de requisitos potenciales de un nuevo sistema o una actualización software. Cada caso de uso proporciona uno o más escenarios que indican cómo debería interactuar el sistema con el usuario o con otro sistema para conseguir un objetivo específico.

Un diagrama de casos de uso es un grafo de actores, conjunto de casos de usos encerrados por los límites de un sistema (rectángulo), asociaciones entre los actores y los casos de uso y generalización entre los actores. Los diagramas de casos de uso muestran elementos procedentes del modelo de casos de uso (casos de uso, actores).

El diagrama de la figura 5, muestra dos actores (El usuario y El Administrador), siete casos de uso (Alta de Usuario, Autentificación, Almacenar Dominio, Modificar Dominio, Eliminar Dominio, Otorgar Atributos, Baja de Usuario), cinco dependencias <<include>> y las asociaciones entre los actores y los casos de uso.

Todos los casos de uso Almacenar Dominio, Modificar Dominio, Eliminar Dominio, Otorgar Atributos, Baja de Usuario, necesitan incluir que el usuario se ha identificado en el sistema. Este comportamiento puede ser extraído a una nueva inclusión de caso de uso llamado Autentificación, por lo cual los cinco casos de uso lo <<include>> (incluyen).

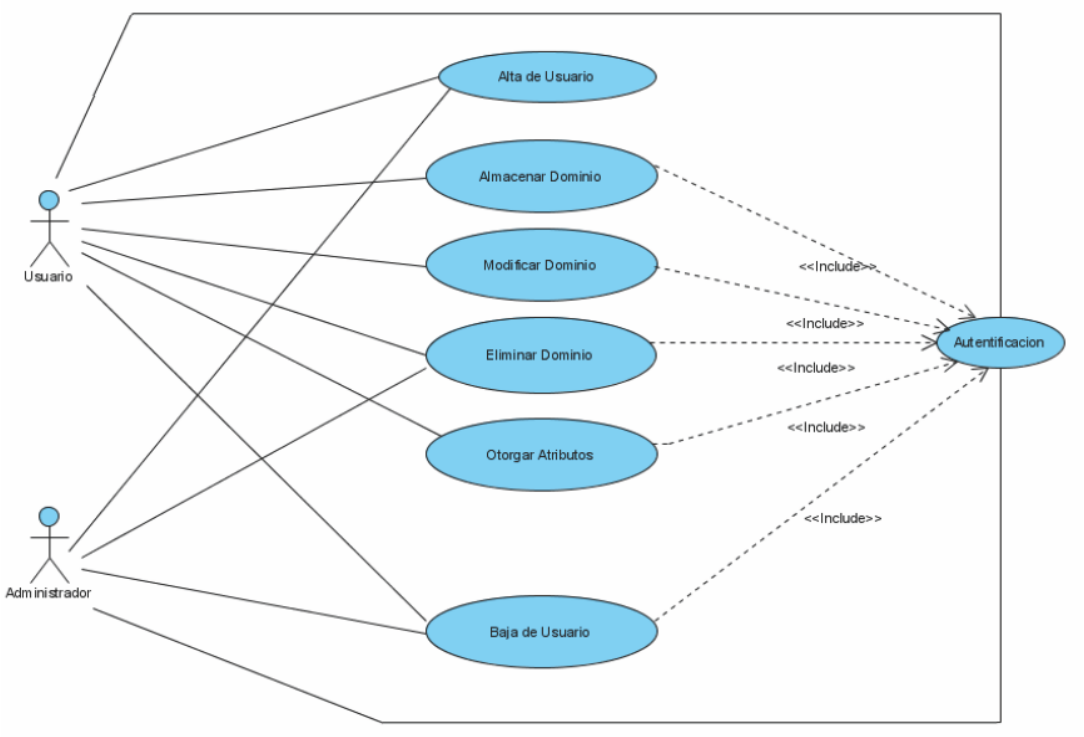

Figura 5. Casos de Uso

Actores

- Usuario: Este actor representa a una persona la cual puede estar o no registrada en el sistema, y este puede ser uso del sistema.

- Administrador: Este actor representa la persona responsable por el mantenimiento del sistema y el control de los usuarios, así como a los datos de estos.

Casos de uso

Alta de Usuario
Especificación de Alto Nivel
Caso de uso: Alta de usuario
Actores: usuario
Descripción: El usuario ingresa a la página principal y da click sobre el link Alta de usuario.

Especificación Extendida
Curso típico de eventos

Acción del actor	Respuesta del sistema
1. El usuario ingresa a la página principal.	

2. El usuario da click sobre el link Alta de usuario.	
	3. Muestra el formulario de alta.
4. El usuario llena el formulario.	
	5. El sistema revisa que los campos están llenos y revisa si no hay un usuario existente con el mismo identificador (ID).
	6. El sistema da de alta al nuevo usuario y muestra un mensaje de operación exitosa.

Acciones Alternativas
Línea 5. El formulario esta incompleto, manda mensaje de falta de parámetros, o el identificador de usuario existe, manda mensaje de poner diferente el identificador.

Autentificación
Especificación de Alto Nivel
Caso de uso: Autentificación
Actores: usuario, administrador
Descripción: El usuario o administrador ingresa a la página principal y da click sobre el link Inicio de Sesión.

Especificación Extendida
Curso típico de eventos

Acción del actor	Respuesta del sistema
1. El usuario o administrador ingresa a la página principal.	
2. El usuario da click sobre el link Inicio de Sesión.	
	3. Muestra una tabla con los campos de ID y Password.
4. El usuario o administrador llena la tabla.	
	5. El sistema revisa que los campos están llenos y revisa si el usuario existe en el sistema.
	6. El sistema muestra la pagina principal del usuario.

Acciones Alternativas
Línea 5. La tabla esta incompleta, manda mensaje de falta de parámetros, o el usuario no existe o el password es incorrecto, manda mensaje de usuario o password incorrecto.

Almacenar Dominio

Especificación de Alto Nivel
Caso de uso: Almacenar Dominio
Actores: usuario
Descripción: El usuario una vez en su página principal selecciona la opción Agregar Direcciones.

Especificación Extendida
Curso típico de eventos

Acción del actor	Respuesta del sistema
1. El usuario ingresa a su página principal.	
2. El usuario selecciona la opción Agregar Direcciones.	
	3. Muestra el formulario de agregar el dominio.
4. El usuario llena el formulario.	
	5. El sistema guarda los datos en la base de datos y muestra un mensaje de operación exitosa.
	6. El sistema después de unos segundos muestra todos los dominios guardados por el usuario.

Modificar Dominio

Especificación de Alto Nivel
Caso de uso: Modificar Dominio
Actores: Usuario
Descripción: El usuario una vez en su página principal selecciona la opción de modificar dominio (indicado por el icono), el cual se muestra al lado izquierdo de cada dominio agregado.

Especificación Extendida
Curso típico de eventos

Acción del actor	Respuesta del sistema
1. El usuario ingresa a su página principal.	
2. El usuario selecciona la opción modificar dominio, en el icono .	
	3. El sistema muestra los datos del dominio para que estos puedan ser modificados por el usuario.
4. El usuario modifica la información	

necesaria.

	5. El sistema guarda las modificaciones en la base de datos y muestra un mensaje de operación exitosa.
	6. El sistema después de unos segundos muestra todos los dominios guardados por el usuario..

Eliminar Dominio

Especificación de Alto Nivel
Caso de uso: Eliminar Dominio
Actores: Usuario
Descripción: El usuario una vez en su página principal selecciona la opción de eliminar dominio(indicado por el icono ), el cual se muestra al lado izquierdo de cada dominio agregado.

Especificación Extendida
Curso típico de eventos

Acción del actor	Respuesta del sistema
1. El usuario ingresa a su página principal.	
2. El usuario selecciona la opción eliminar dominio, en el icono  .	
	3. El sistema elimina los datos del dominio seleccionado y muestra un mensaje de operación exitosa.
	4. El sistema después de unos segundos muestra todos los dominios guardados por el usuario..

Baja de Usuario

Especificación de Alto Nivel
Caso de uso: Baja de usuario
Actores: usuario
Descripción: El usuario ingresa a la página principal y da click sobre el link Baja del Sistema.

Especificación Extendida
Curso típico de eventos

Acción del actor	Respuesta del sistema
1. El usuario ingresa a su página	

principal.

2. El usuario da click sobre el link Baja del Sistema.	
	3. El sistema muestra un mensaje de eliminación del usuario del sistema y pide la confirmación.
4. El usuario selecciona la opción de proseguir con la baja del sistema.	
	5. El sistema da de baja al usuario del sistema y muestra un mensaje de baja del usuario.
	6. El sistema después de unos segundos muestra la pagina principal del sistema.

Acciones Alternativas
Línea 5. El usuario selecciona la opción de cancelar la baja del sistema. El sistema cancela la baja y muestra todos los dominios guardados por el usuario.

CAPITULO 3

DISEÑO DE LA BASE DE DATOS

3.1 DISEÑO CONCEPTUAL DE LA BASE DE DATOS

En esta etapa se construyó un esquema de la información para tener mas claro lo que contendrá nuestro sistema, de esta manera se describen los datos así como, atributos y relaciones.

3.1.1 Modelo Entidad-Relación

Los Diagramas Entidad-Relación describen el esquema de una base de datos. Las Entidades representan objetos reales, están representadas mediante rectángulos. Los Atributos representan propiedades de estos objetos reales, están representados mediante óvalos. Las Relaciones representan los enlaces entre las entidades, están representadas mediante rombos.

Siguiendo el análisis de la información o requerimientos, la base de datos quedo constituida por dos entidades, las cuales tienen una cardinalidad de uno a muchos, un usuario puede guardar uno a muchos URL's, y un URL puede ser guardado por un usuario, esto se muestran en la figura 6.

El proyecto contará con la siguiente información organizada en tablas, que a continuación se explican:

- USUARIO
- URL

TABLA USUARIO: La tabla usuario contendrá toda la información concerniente a cada uno de los usuarios del sistema.

ID	Identificador para cada usuario.
NOMBRE	Nombre del usuario.
MAIL	Correo electrónico del usuario.
PASSWORD	Password del usuario para que este pueda ingresar al sistema.

TABLA URL: La tabla url contendrá información concerniente a los lugares de interés del usuario.

IDENTIFICADOR	Identificador para cada dominio.
DESCRIPCION	Nombre de referencia del dominio.
PERMISO	Atributo de acceso, puede ser Público o Privado.
DIRRECCION	Dirección del dominio.
TIPO	Clase de dominio (Informativo, Entretenimiento, Personal, Ocio, Otros).

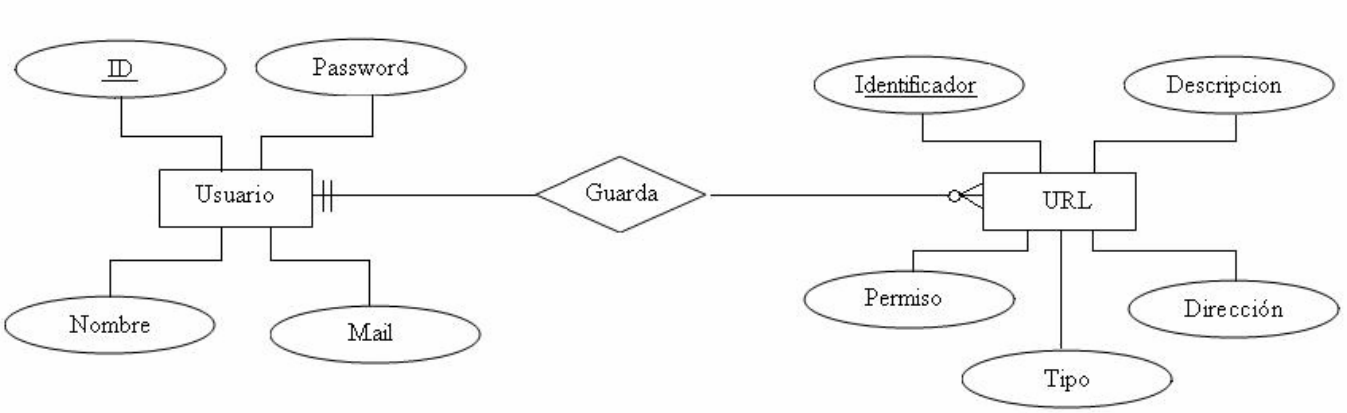

Figura 6. Modelo Entidad-Relación

3.1.2 Descripción de Entidades y Relaciones

Una vez tomados los datos correspondientes, procedemos a hacer el diagrama Entidad – Relación, en el cual se mostrará la relación entre los datos desde el punto de vista del usuario.

La relación que se toma, es la de la entidad del usuario con el URL, la cual tiene cardinalidad de uno a muchos (1: m), ya que un usuario puede guardar una o más URL's, y un URL puede ser guardado por un usuario. La relación entre ambas tablas se llamará **Guarda**, ver tabla 1.

Entidad	**Atributos**	**Cardinalidad**	**Entidades con las que se relaciona**
usuario	ID, Password, Mail, Nombre.	UNO A VARIOS	URL
URL	Identificador, Descripción, Permiso, Tipo, Dirección.	VARIOS A UNO	usuario

Tabla 1. Descripción

3.1.3 Diseño Lógico

Convertir el esquema conceptual en el esquema lógico se eliminan del esquema conceptual las estructuras de datos que los sistemas relacionales no modelan directamente:

- Eliminar las relaciones de muchos a muchos, sustituyendo cada una de ellas por una nueva entidad y dos relaciones de uno a muchos de esta nueva entidad con las entidades originales.

- Eliminar las relaciones entre tres o más entidades, sustituyendo cada una de ellas por una nueva entidad (débil) intermedia que se relaciona con cada una de las entidades originales. La cardinalidad de estas nuevas relaciones binarias dependerá de su significado

- Eliminar las relaciones recursivas, sustituyendo cada una de ellas por una nueva entidad (débil) y dos relaciones binarias de esta nueva entidad con la entidad original. La cardinalidad de estas relaciones dependerá de su significado.

- Eliminar las relaciones con atributos, sustituyendo cada una de ellas por una nueva entidad. La cardinalidad de estas relaciones dependerá del tipo de la relación original y de su significado.

- Eliminar los atributos multievaluados, sustituyendo cada uno de ellos por una nueva entidad y una relación binaria de uno a muchos con la entidad original.

- Revisar las relaciones de uno a uno, ya que es posible que se hayan identificado dos entidades que representen el mismo objeto (sinónimos). Si así fuera, ambas entidades deben integrarse en una sola.

- Eliminar las relaciones redundantes. Una relación es redundante cuando se puede obtener la misma información que ella aporta mediante otras relaciones. El hecho de que haya dos caminos diferentes entre dos entidades no implica que uno de los caminos corresponda a una relación redundante, eso dependerá del significado de cada relación.

Finalmente el esquema conceptual queda convertido en el esquema lógico, ver figura 7.

Usuario

ID	Password	Nombre	Mail

URL

Identificador	Tipo	Descripción	Permiso	Dirección	ID(FK)

Figura 7. Esquema Lógico

3.1.4 Normalización

El proceso de normalización tiene un nombre y una serie de reglas para cada fase. Una ventaja de la normalización de la base de datos es el consumo de espacio. La tabla 2 muestra la base de datos del proyecto.

ID	Password	Nombre	E-Mail	Identificador	Tipo	Descripcion	Permiso	Direccion

Tabla 2. Datos de la Base de Datos

La Primera Forma Normal establece que las columnas solo deben de tener datos atómicos. La tabla 3 esta en la primera forma normal.

ID	Password	Nombre	E-Mail	Identificador	Tipo	Descripcion	Permiso	Direccion

Tabla 3. Primera Forma Normal

La Segunda Forma Normal establece que todas las dependencias parciales se deben eliminar y separar dentro de sus propias tablas. Una dependencia parcial es un término que describe a aquellos datos que no dependen de la clave de la tabla para identificarlos. La tablas 4 y 5 son el resultado de la segunda forma normal.

ID	Password	Nombre	E-Mail

Tabla 4. Tabla Usuarios.

URL	URLLID	Identificador	Tipo	Descripción	Permiso

Tabla 5. Tabla URL

La Tercera Forma Normal señala que hay que eliminar y separar cualquier dato que no sea clave. La tabla de la tercera forma normal queda de la misma forma como se muestra en las Tablas 4 y 5, ya que no se repiten datos.

La normalización es una ciencia subjetiva. Determinar las necesidades de simplificación depende de la persona. Si la base de datos va a proveer información a un solo usuario para un propósito simple y existen pocas posibilidades de expansión, normalizar sus datos hasta la 3FN sea quizá algo extremoso. Las reglas de normalización existen como guías para crear tablas que sean fáciles de manejar, así como flexibles y eficientes.

3.2 DISEÑO DE LA APLICACIÓN

3.2.1 Diagramas de Estado, Secuencia y Colaboración

Alta de usuario

El sistema muestra un formulario para que el usuario ingrese sus datos personales: nombre completo, dirección de correo, en caso de que falte algún dato, nuevamente el sistema muestra el formulario y pide a este que lo ingrese, una ves que los datos están completos el sistema comprueba que el ID agregado por el usuario este disponible, en caso de que ya exista pide al usuario que ingrese uno nuevo, en caso de que este disponible comprueba que el Password y la comprobación del Password sean iguales, en caso de no serlo pide nuevamente el Password y su confirmación, una vez que sea idéntico el sistema ya puede dar de alta al nuevo usuario, ver figuras 8, 9, 10.

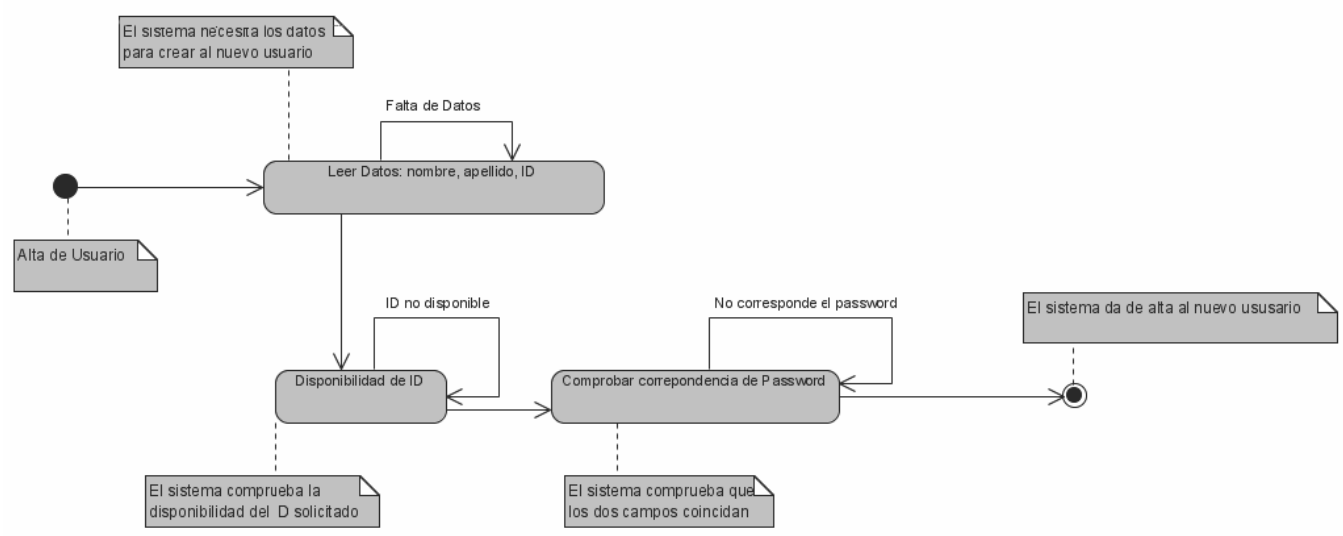

Figura 8. Diagrama de Estado de Alta de Usuario

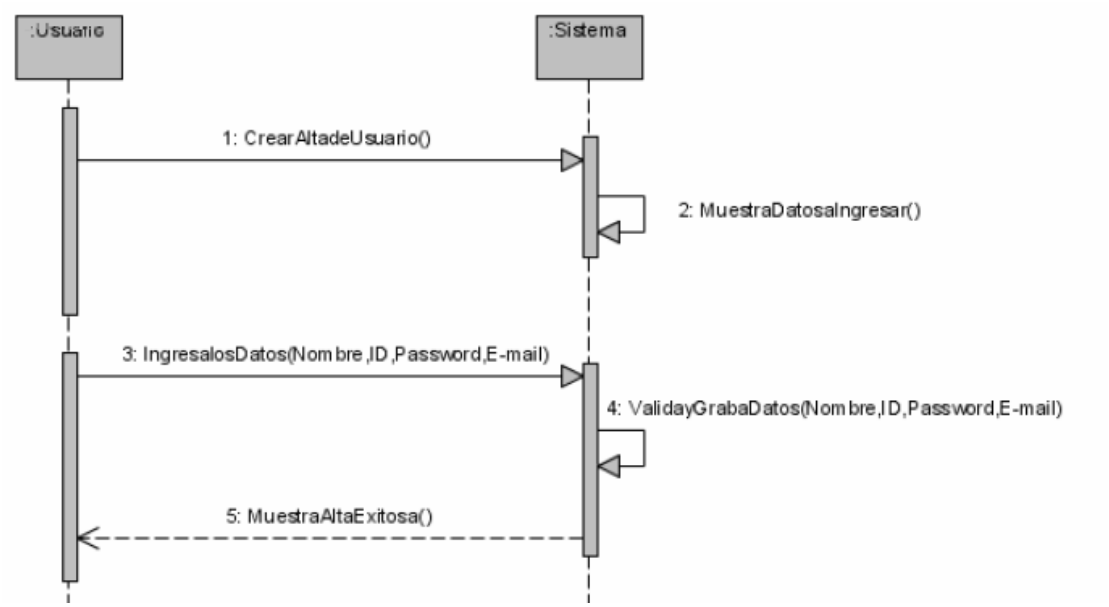

Figura 9. Diagrama de Secuencia de Alta de Usuario

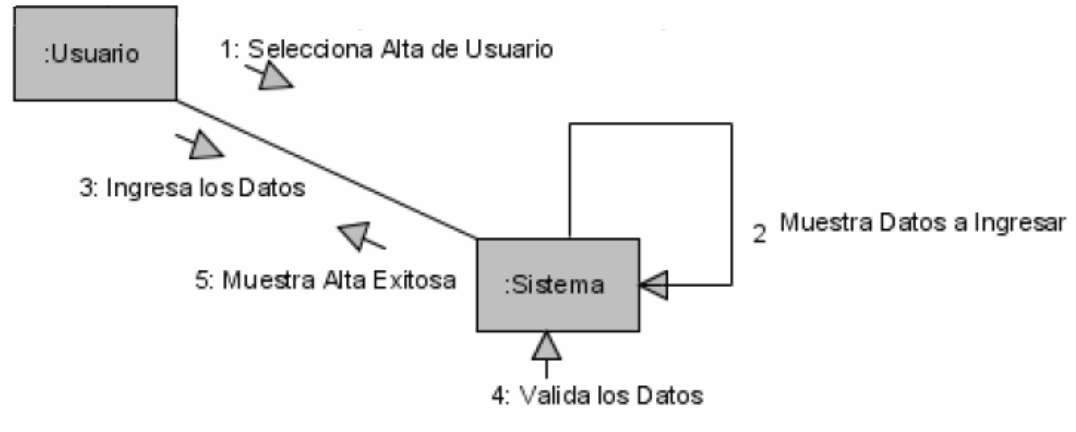

Figura 10. Diagrama de Colaboración de Alta de Usuario

Autentificación

El sistema exige que cada usuario se autentifique para que este pueda hacer uso de este, y así poder manipular los datos de la cuanta de usuario. El sistema muestra dos campos, son el ID y el Password, una vez que el usuario los ingresa, el sistema comprueba que estos datos sean correctos y que pertenezcan al usuario, en caso de no serlo el sistema pide nuevamente los datos ID y Password, en caso de ser correctos el sistema permite al usuario ingresar, ver figura 11, 12, 13.

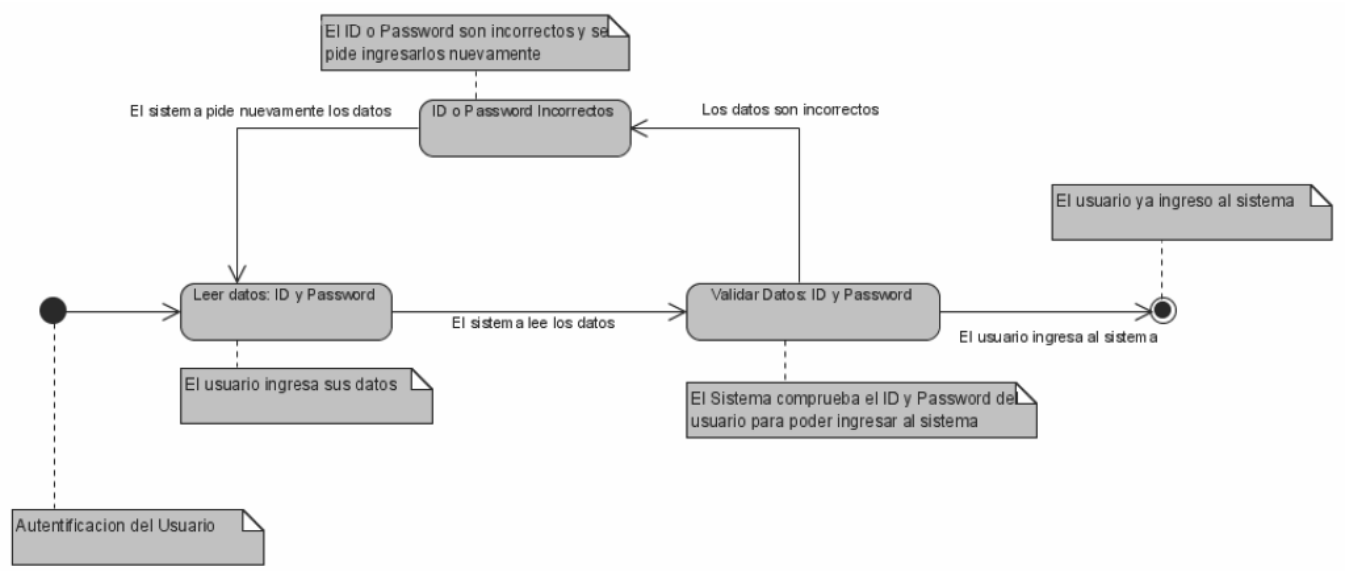

Figura 11. Diagrama de Estado de Autentificación

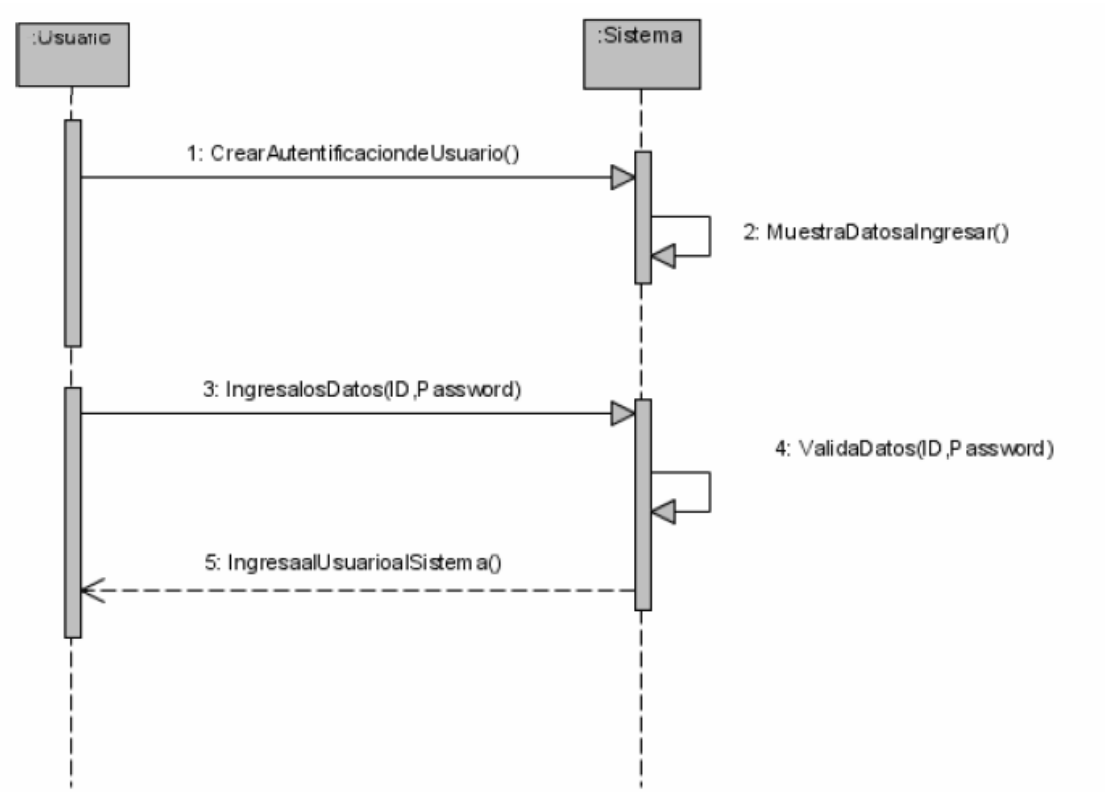

Figura 12. Diagrama de Secuencia de Autentificación

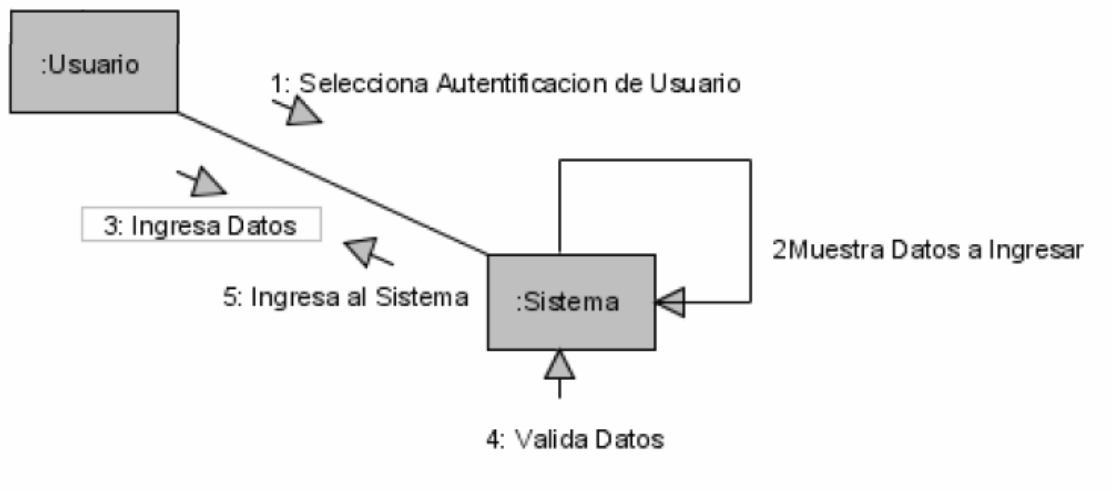

Figura 13. Diagrama de Colaboración de Autentificación

Baja de usuario

El sistema permite a cualquier usuario darse de baja del sistema en cualquier momento, y este pide una confirmación del usuario para proseguir con la baja. El sistema muestra los campos ID y Password, una vez que el usuario los ingresa, el sistema comprueba que estos datos sean correctos y que pertenezcan al usuario, en caso de no serlo el sistema pide nuevamente los datos, en caso de ser correctos el sistema permite al usuario ingresar, se pide una confirmación de el usuario para que se prosiga con la baja en el sistema, el usuario acepta y se da la baja del usuario, ver figura 14, 15, 16.

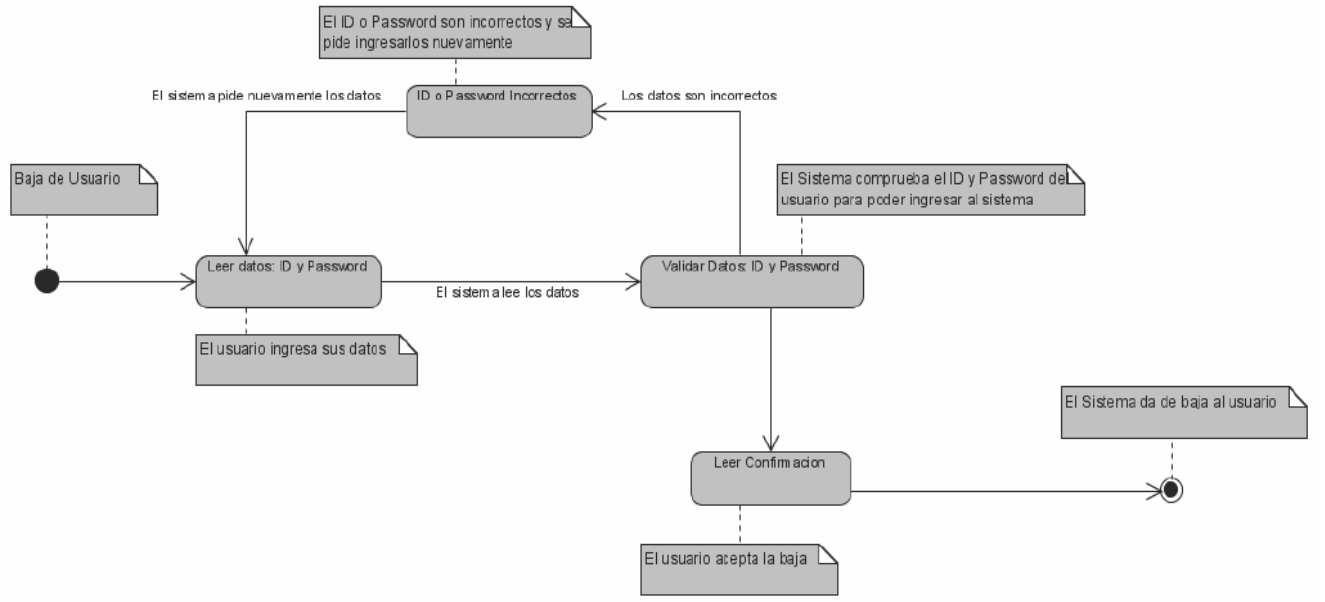

Figura 14. Diagrama de Estado de Baja de usuario.

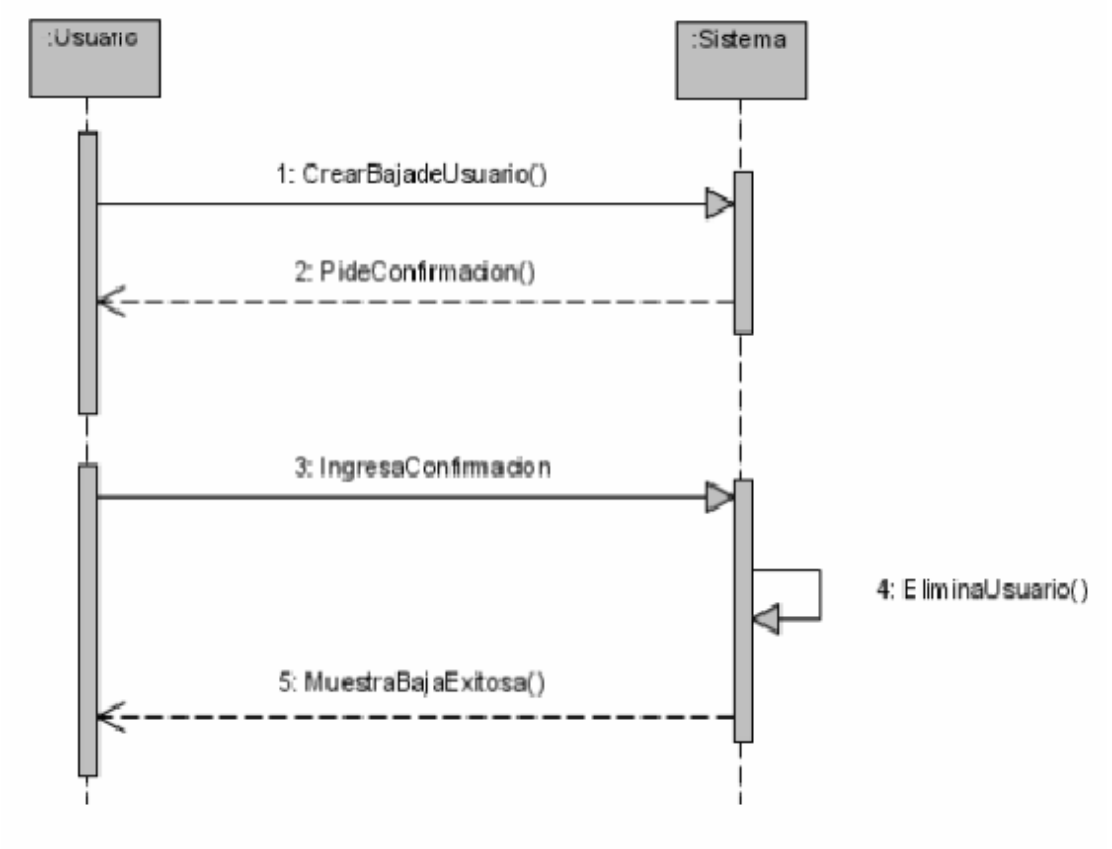

Figura 15. Diagrama de Secuencia de Baja de usuario.

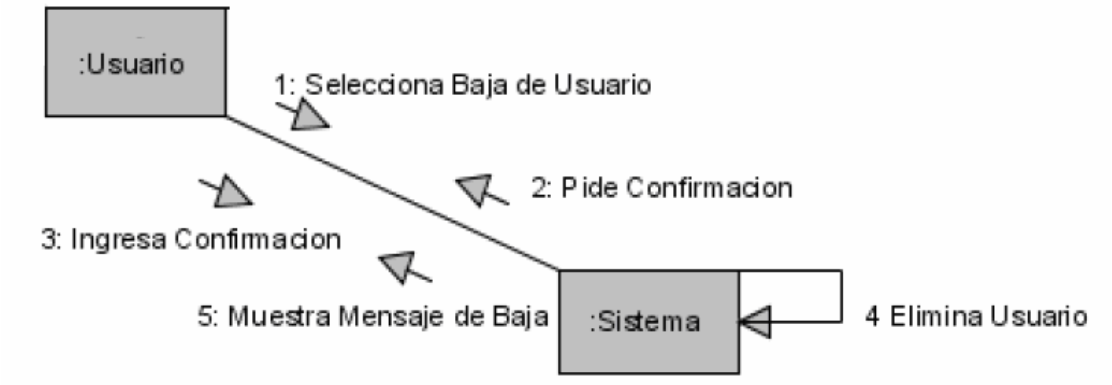

Figura 16. Diagrama de Colaboración de Baja de usuario.

Almacenar dominio

El sistema muestra los campos para agregar un dominio, el usuario ingresa los datos del dominio, el sistema lee los datos y los guarda en la base de datos, ver figura 17, 18, 19.

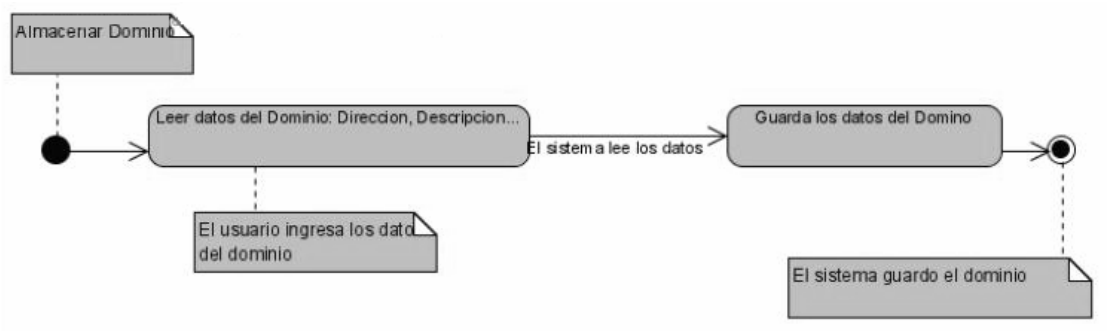

Figura 17. Diagrama de Estado de Almacenar Dominio.

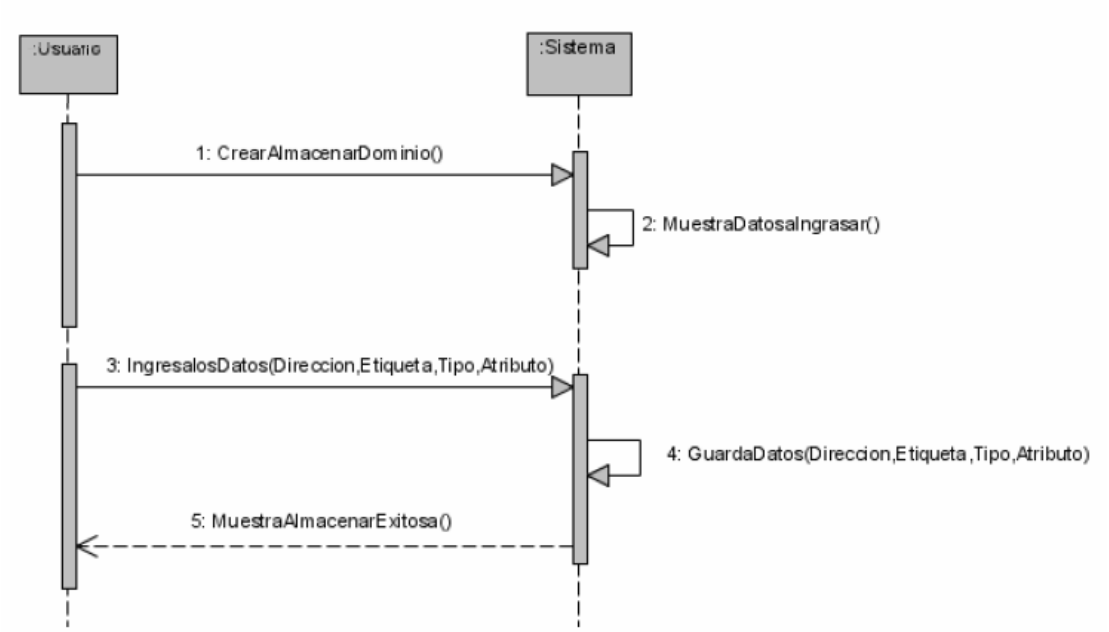

Figura 18. Diagrama de Secuencia de Almacenar Dominio.

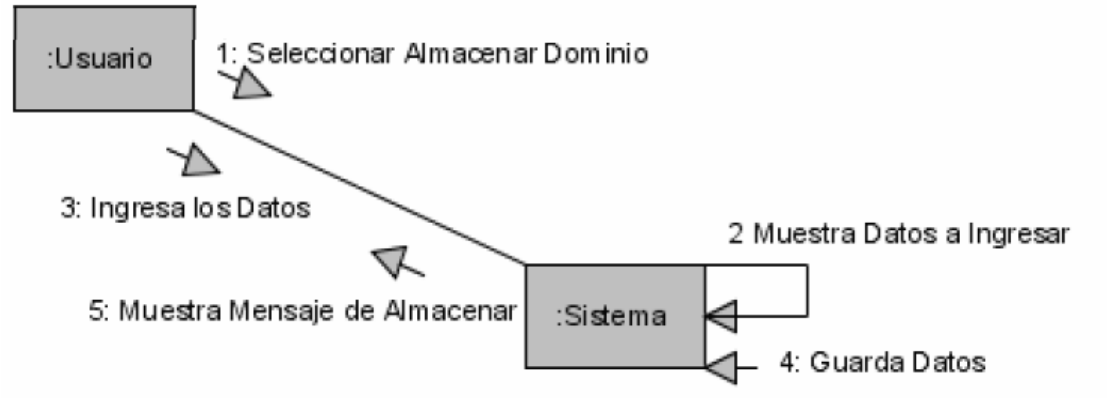

Figura 19. Diagrama de Colaboración de Almacenar Dominio.

Modificar dominio.

El sistema muestra los datos del dominio a modificar, cuando el usuario realice las modificaciones el sistema lee los datos y los guarda en la base de datos, ver figura 20, 21, 22.

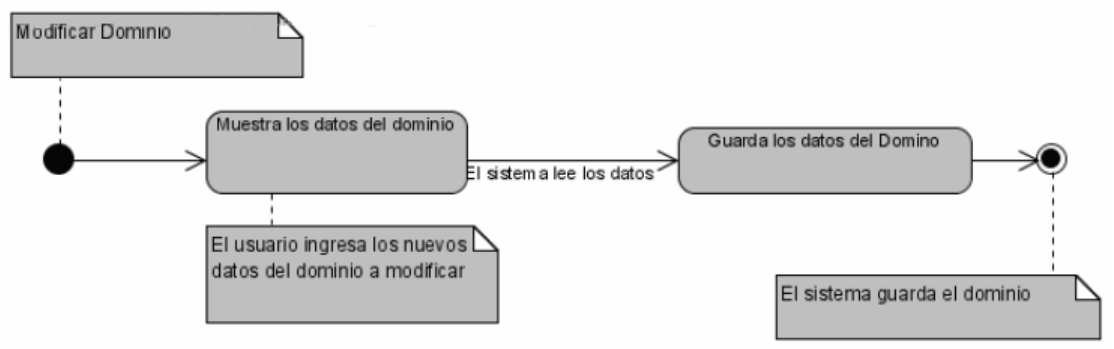

Figura 20. Diagrama de Estado de Modificar Dominio.

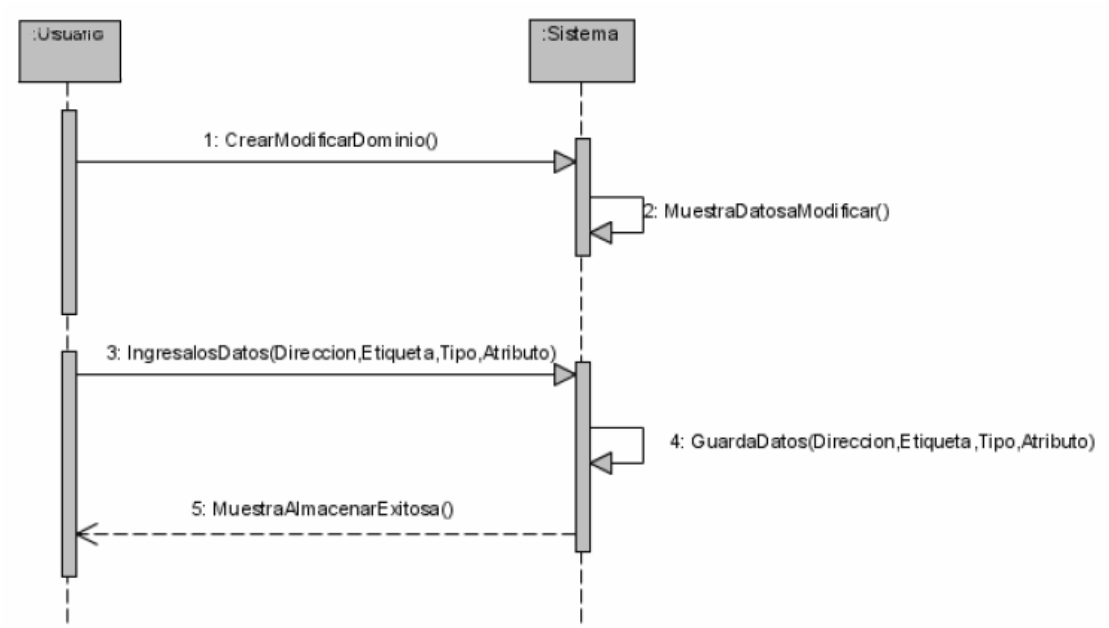

Figura 21. Diagrama de Secuencia de Modificar Dominio.

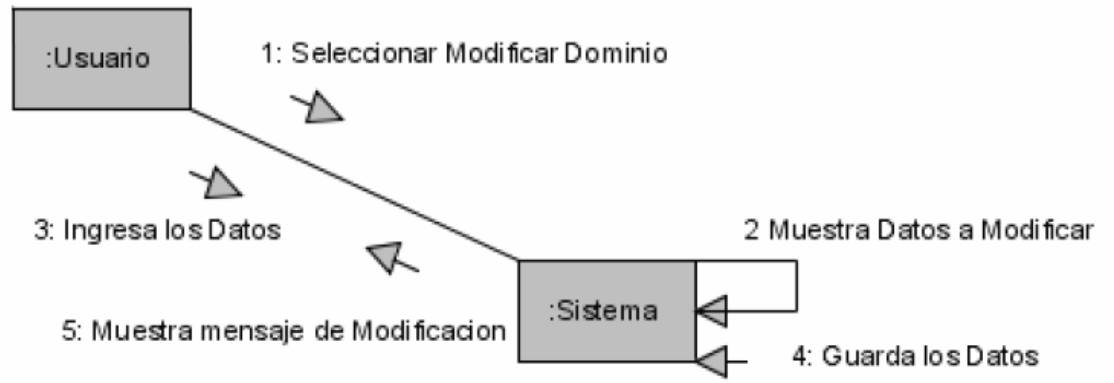

Figura 22. Diagrama de Colaboración de Modificar Dominio.

Eliminar dominio

El sistema muestra todos los dominios guardados por el usuario, el usuario selecciona el dominio a eliminar, el sistema elimina el dominio de la base de datos, ver figura 23, 24, 25.

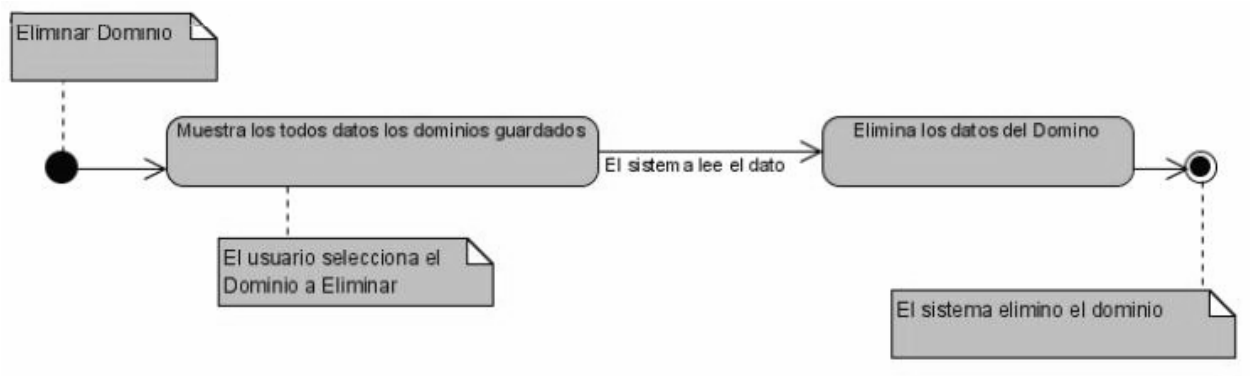

Figura 23. Diagrama de Estado de Eliminar Dominio.

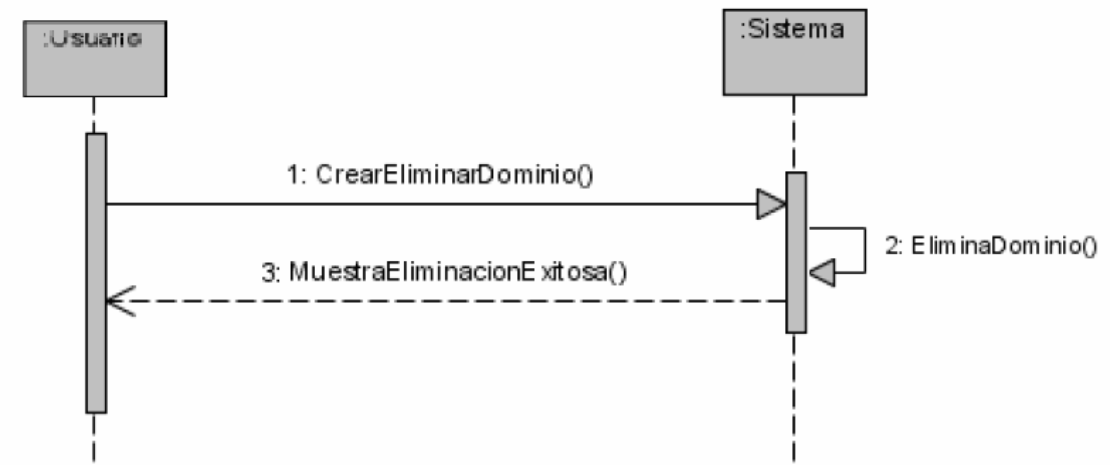

Figura 24. Diagrama de Secuencia de Eliminar Dominio.

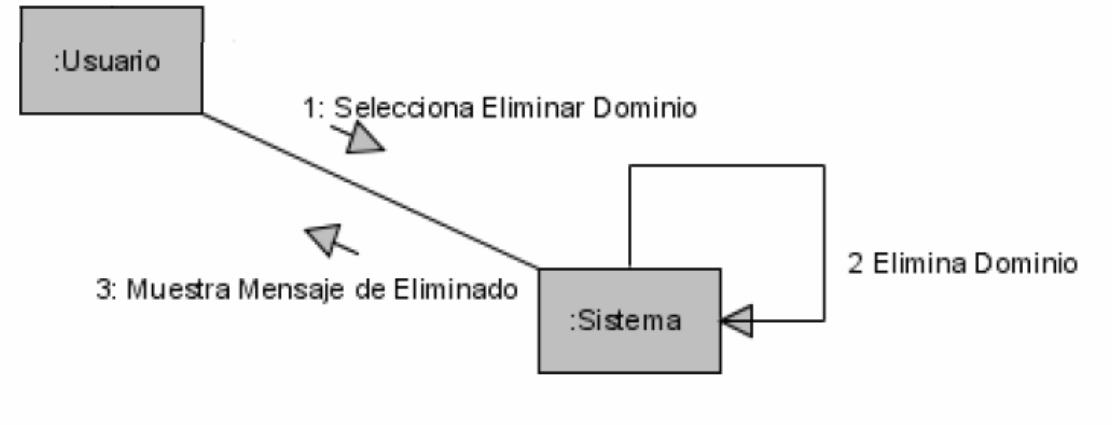

Figura 25. Diagrama de Colaboración de Eliminar Dominio.

3.2.2 Diagrama de Clases

Un diagrama de clases (Figura 26), sirve para visualizar las relaciones entre las clases que involucran el sistema. El diagrama de clases esta compuesto por los siguientes elementos:

Clase Usuario: ID, Password, Mail, Nombre.
Clase URL: Dirección, Etiqueta, Atributo, Tipo.

Relaciones: Composición, Asociación con multiplicidad.

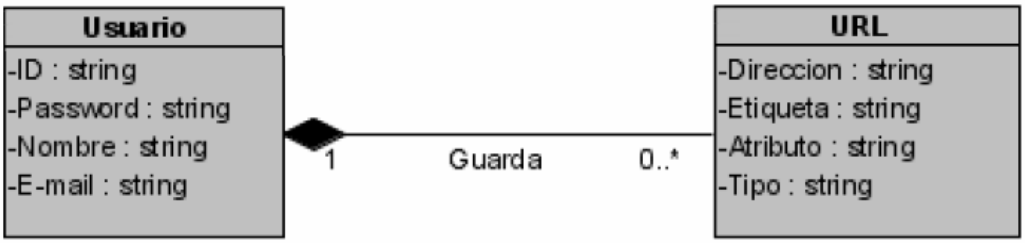

Figura 26. Diagrama de Clases

CAPITULO 4

IMPLEMENTACIÓN Y PRUEBAS

4.1 IMPLEMENTACIÓN

La implementación de la base de datos por medio de la consola gráfica MySQL, en la creación de la Base de Datos y las tablas se realizo mediante código de manera sencilla.

Creación de la Base de Datos (LINK).

```
mysql> CREATE DATABASE LINK;
Query OK, 1 row affected (0.00 sec)

mysql> USE LINK
Database changed
mysql>
```

Una vez creada la base de datos, esta se carga y se crean las tablas en las cuales se almacenara la información correspondiente para cada entidad que se requiera.

La forma de crear una tabla es sencilla, puesto que sólo se le pone el nombre, los campos y los tipos de valor que llevará cada campo de esta, es decir, se le dan todas las propiedades que la tabla requiere.

Creación de la tabla Usuarios (USERS)

```
mysql>CREATE TABLE `USERS` (
  `ID` varchar(20) default NULL,
  `USER` varchar(20) default NULL,
  `PASS` varchar(10) default NULL,
  `MAIL` varchar(30) default NULL
  PRIMARY KEY  (`ID`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
Query OK, 0 rows affected (0.02 sec)
mysql>
```

Creación de la tabla Urls (URLS)

```
mysql>CREATE TABLE `URLS` (
  `urlid` int(11) NOT NULL auto_increment,
  `urllid` varchar(20) default NULL,
  `url` varchar(250) default NULL,
  `type` varchar(15) default NULL,
  `permit` varchar(7) default NULL,
  `descrip` varchar(100) default NULL,
  PRIMARY KEY  (`urlid`)
) ENGINE=MyISAM AUTO_INCREMENT=46 DEFAULT CHARSET=latin1;
Query OK, 0 rows affected (0.02 sec)
mysql>
```

4.2 IMPLEMENTACIÓN DE LA INTERFAZ

4.2.1 Página de Inicio

Como interfaz para ver el diseño, he utilizado el software FireFox, el cual ofrece muy buenas herramientas para la navegación en Páginas Web.

Lo primero que se hace, es la interfaz de la página de inicio, la cual contiene una descripción del propósito de la página. La página se muestra la figura 27.

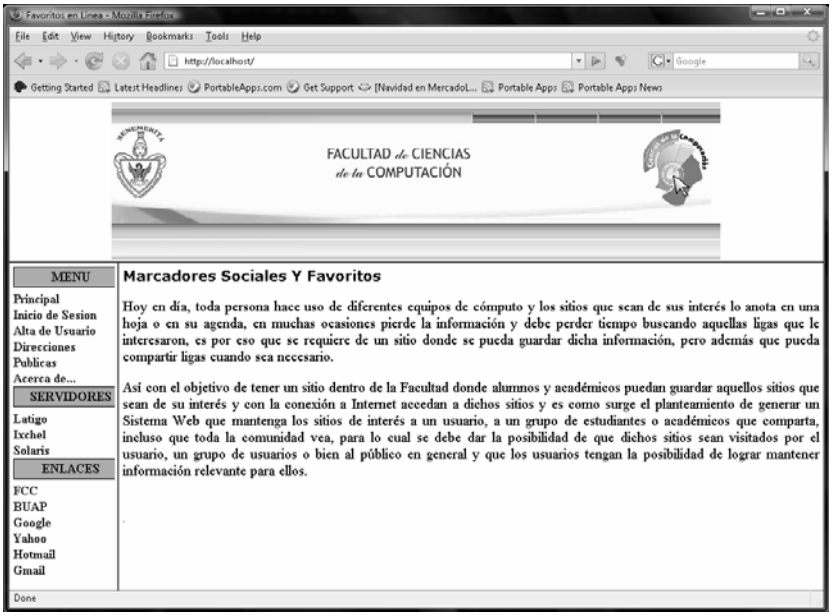

Figura 27. Página de Inicio

En la página de inicio, se observa que se tiene un menú con submenús, los cuales tienen varios links:

En el submenú MENU se encuentran los links:

- Principal: Este link muestra la página principal del portal, la cual se muestra en la figura 27.

- Inicio de Sesión: En esta página el usuario puede autentificarse para que pueda ingresar al sistema y así poder manipular los dominios a su criterio. En caso de que el usuario o el password sean incorrectos manda un mensaje de error “CONTRASEÑA O USUARIO INCORRECTO”. Como muestra en la figura 28.

- Alta de Usuario: En esta página el usuario puede darse de alta. Proporcionando sus datos personales: nombre completo, dirección de correo. Como también los datos que se van a utilizar en el sistema: su ID y su password, estos dos últimos para poder ingresar al sistema. En esta

página se muestra un Acuerdo de condiciones de Uso del portal. En caso de que falte algún campo, el usuario ya existe, o no coinciden el password, el portal le manda un mensaje de error “ALGUN CAMPO ESTA VACIO” o “ID NO DISPONIBLE” o “LAS CONTRASEÑAS NO COINCIDEN” dependiendo de la situación. ”. Como muestra en la figura 29.

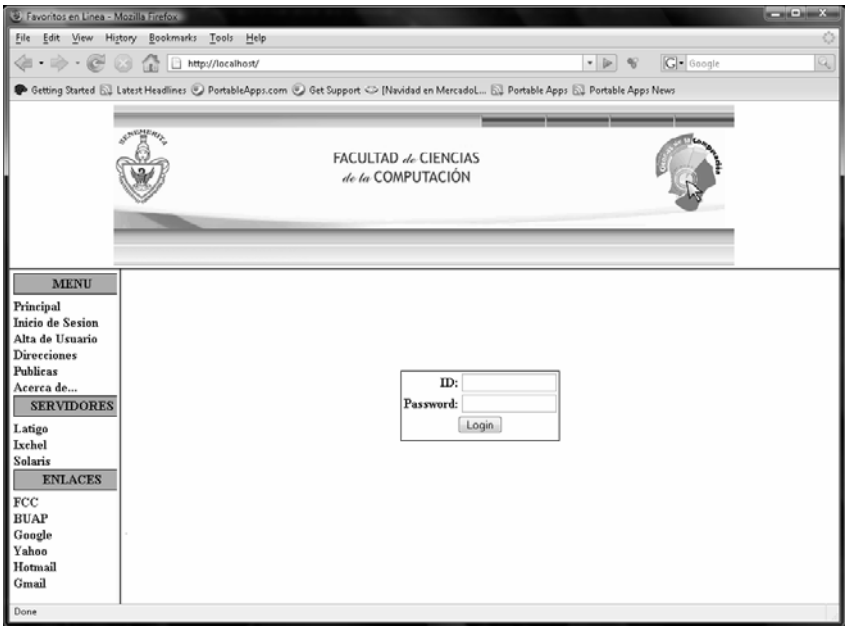

Figura 28. Inicio de Sesión

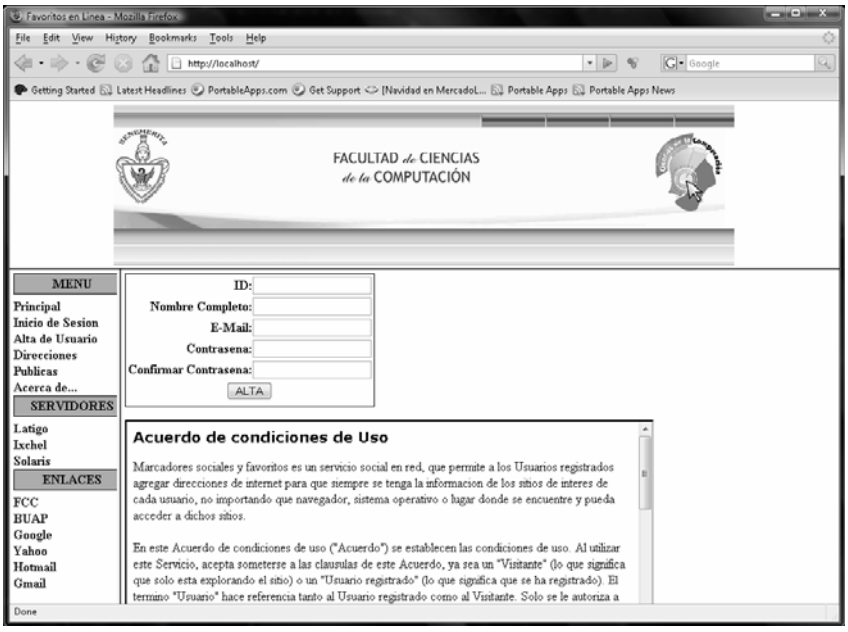

Figura 29. Alta de Usuario

- Direcciones Públicas: En esta página cualquier usuario puede ver todas las direcciones públicas de todos los usuarios, este puede o no estar en el sistema. Como muestra en la figura 30.

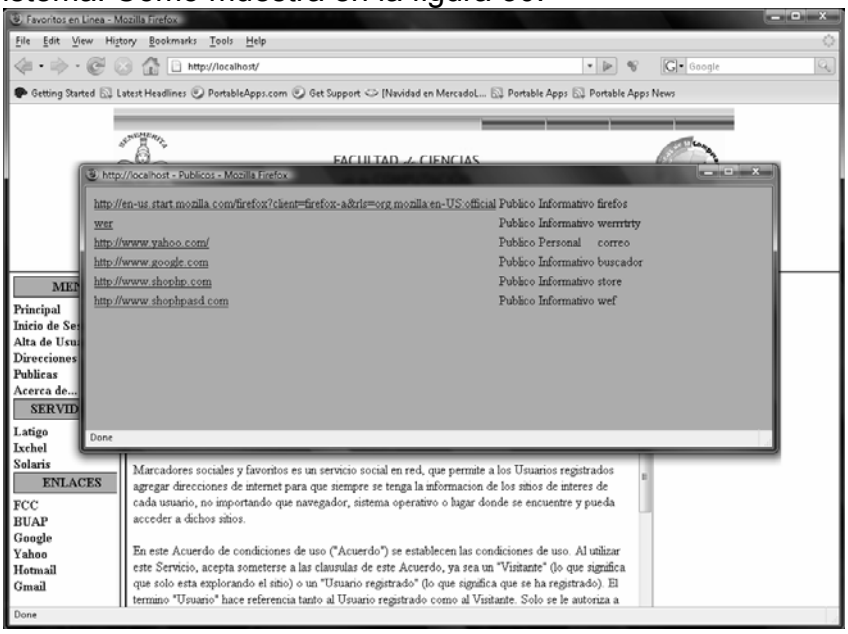

Figura 30. Direcciones Públicas

En el submenú SERVIDORES se encuentran los links:

- Latigo: Este link muestra la página principal del servidor LATIGO de la facultad de computación. Uno de los más usados por los alumnos.

- Ixchel: Este link muestra la página principal del servidor IXCHEL de la facultad de computación. Uno de los mas usados por los alumnos y por los maestros.

- Solarium: Este link muestra la página principal del servidor de correos SOLARIUM de la facultad de computación. Uno de los más usados por los maestros.

En el submenú ENLACES se encuentran los links:

- FCC: Este link muestra la página principal del portal de la Facultad Ciencias de Computación.

- BUAP: Este link muestra la página principal del portal de la Benemérita Universidad Autónoma de Puebla.

- Google: Este link muestra la página google el motor de búsqueda en Internet más grande y más usado. Ofrece una forma rápida y sencilla de encontrar información en la Web.

- Yahoo: Este link muestra la página Yahoo, Posee un portal de Internet, un directorio Web y una serie de servicios, incluido el popular correo electrónico Yahoo!.

- Hotmail: Este link muestra la página Hotmail, un proveedor de correo electrónico tanto gratuito como de pago. Tiene presencia en gran parte del mundo. Actualmente pertenece a la red MSN, la cual a su vez pertenece a la empresa Microsoft.

- Gmail: Este link muestra la página Gmail, es un servicio de correo electrónico y POP3 gratuito, proporcionado por la empresa estadounidense Google.

4.2.1 Interfaz del Usuario

Una vez ingresado el usuario con su ID y password, entrará a la página principal de los usuarios, en la cual se muestran las opciones con las que cuenta como usuario, ver figura 31.

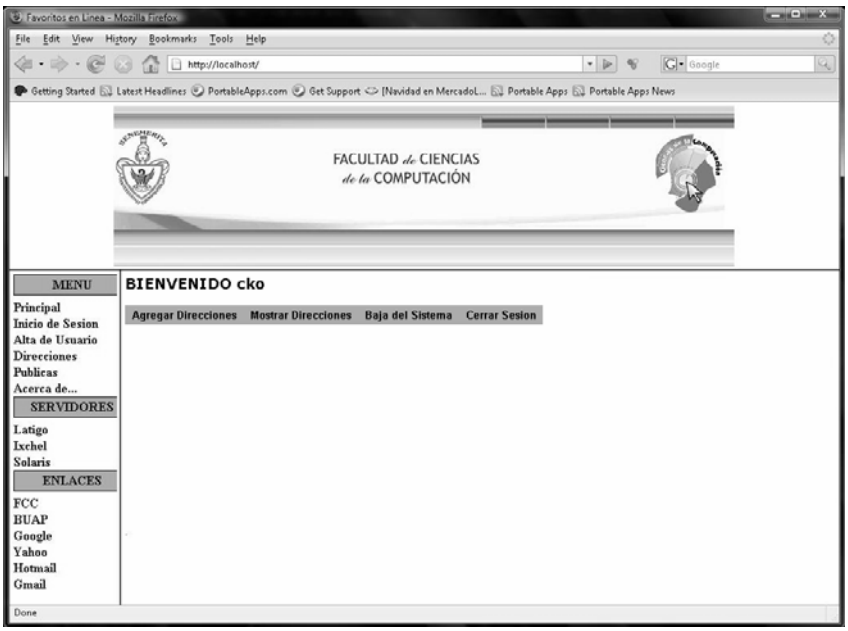

Figura 31. Página Inicio de Usuario

Entre las opciones que tiene, están:

- Agregar Direcciones: En esta opción el usuario agrega las direcciones de su interés, para la organización de estas debe agregar URL (el cual es la dirección en Internet), Descripción (es una nota de referencia de la dirección agregada), el tipo de dirección (puede ser de 5 tipos: Informativo, Entretenimiento, Personal, Ocio u Otros), atributo (Publico o Privado). Cuando se llenan los campos y se presiona el botón Agregar, los datos son agregados a la base de datos y la página muestra un

mensaje “URL: (Dirección) Agregado” por 2 segundos y después muestra las direcciones agregadas por el usuario, ver figura 32.

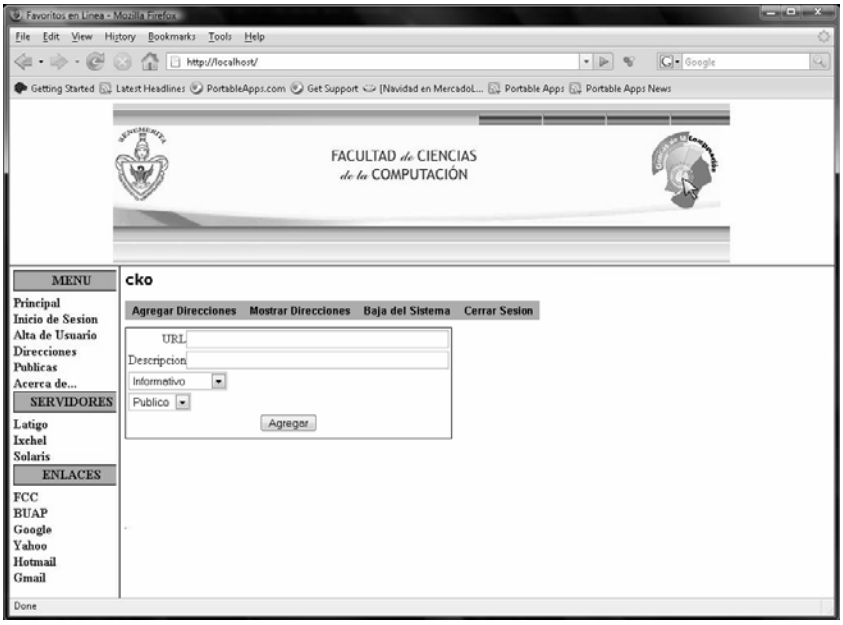

Figura 32. Página Agregar Direcciones

- Mostrar Direcciones: Esta opción muestra las direcciones agregadas por el usuario, hay dos opciones para las direcciones, editar o eliminar , si se presiona sobre la imagen editar, muestra la página de agregar direcciones pero con los campos llenos con la información correspondiente, solo se utiliza para la modificación de estas, y en caso de presionar sobre la imagen de eliminar, la dirección es eliminada de la base de datos y la página muestra un mensaje “Eliminó URL Satisfactoriamente” por 2 segundos y regresa a la página mostrar direcciones, ver figura 33.

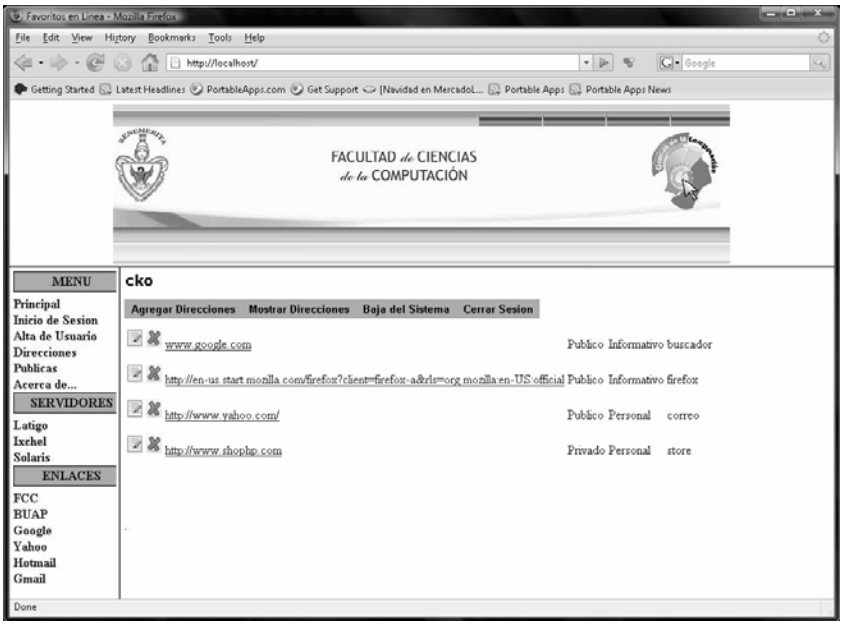

Figura 33. Página Mostrar Direcciones

- Baja del Sistema: Esta opción muestra una confirmación para que el usuario pueda darse de baja del sistema. En caso de que acepte el usuario es borrado de la base de datos y muestra la página principal del portal, y en caso contrario se muestra la página mostrar direcciones del usuario, ver figura 34.

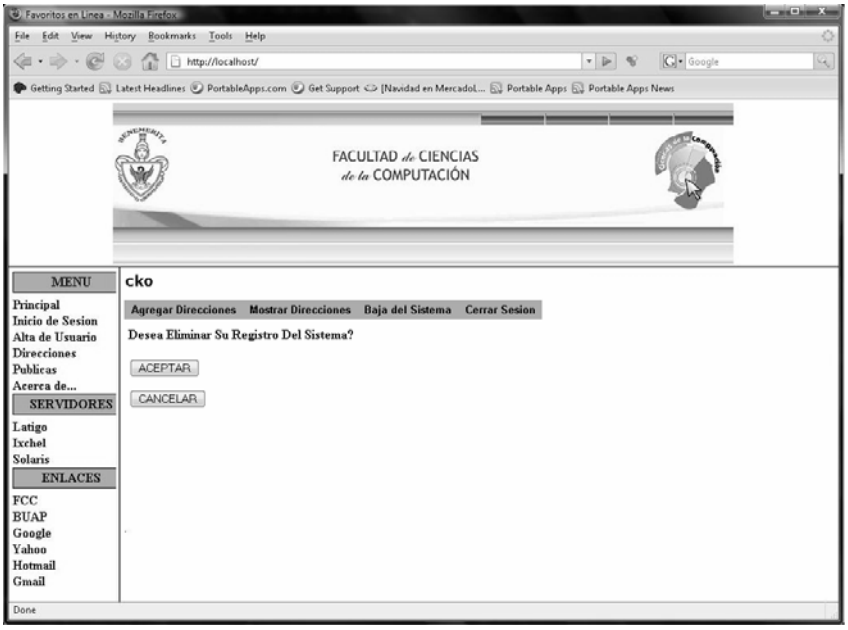

Figura 34. Página Baja del Sistema

- Cerrar Sesión: Esta opción se utiliza para salir de la sesión actual. Muestra un mensaje “HA CERRADO LA SESION CORRECTAMENTE” y después de unos segundos muestra la página principal del portal, ver figura 35.

Figura 35. Página Cerrar Sesión

4.2.2 Interfaz del Administrador

Una vez ingresado el administrador con su ID y password, entrará a la página principal del administrador, en la cual se muestran las opciones con las que cuenta como administrador, ver figura 36.

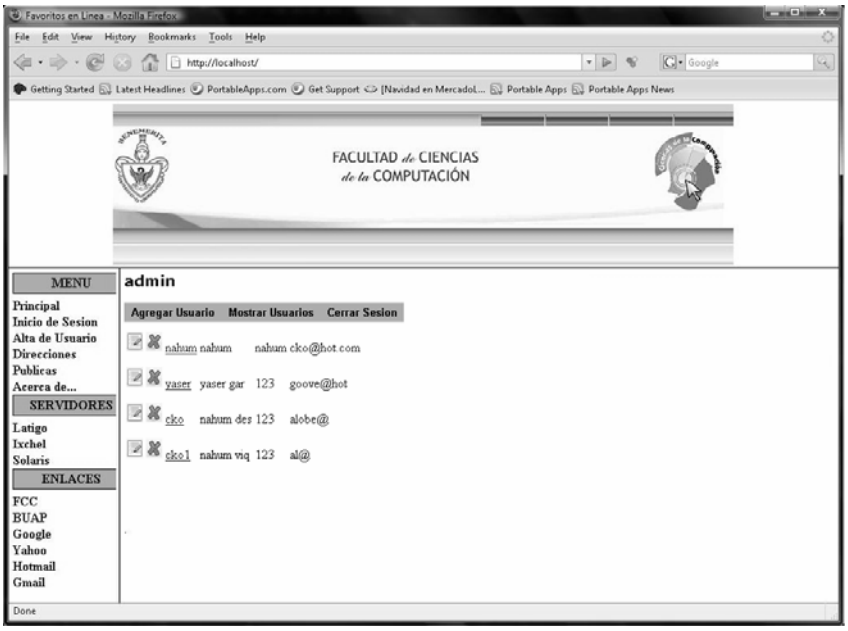

Figura 36. Página Inicio del Administrador

Entre las opciones que tiene, están:

- Agregar Usuario: En esta opción el administrador puede agregar manualmente a los usuarios, debe agregar ID (el identificador del usuario), Nombre, E-mail y Password. Cuando se llenan los campos y se presiona el botón Agregar, los datos son agregados a la base de datos y

la página muestra un mensaje “USUARIO: (nombre del usuario) Agregado” por 1 segundo y después muestra los usuarios del sistema, ver figura 37.

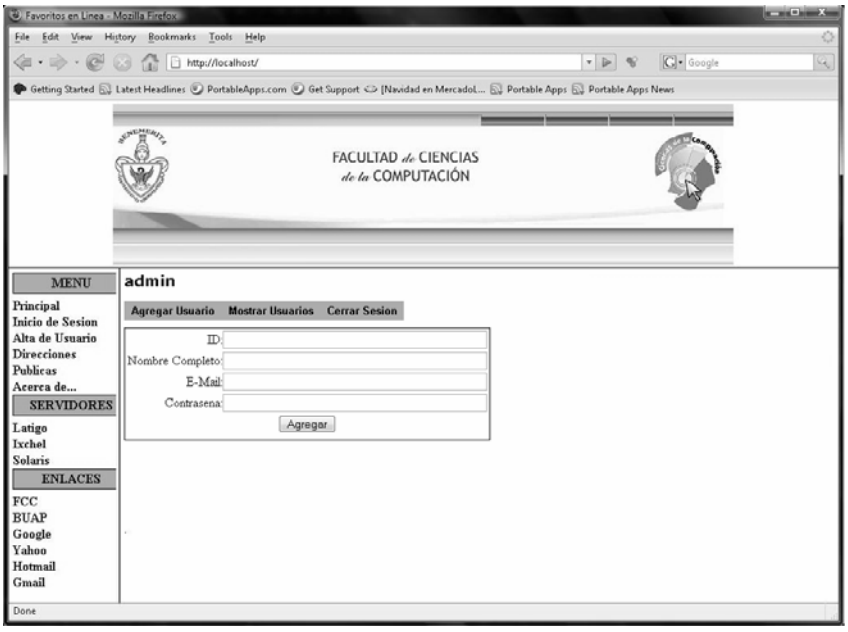

Figura 37. Página Agregar Usuario

- Mostrar Usuarios: Esta opción muestra los usuarios que tiene el sistema, ver figura 38, hay tres opciones, editar  , eliminar  , o mostrar las direcciones del usuario, este ultimo presionando sobre el usuario, ver figura 39, si se presiona sobre la imagen editar, muestra la página de agregar usuario pero con los campos llenos con la información correspondiente, solo se utiliza para la modificación de éstas, y en caso de presionar sobre la imagen de eliminar, el usuario es eliminado de la base de datos y la página muestra un mensaje “Elimino usuario Satisfactoriamente” por unos segundos y regresa a la página mostrar usuarios.

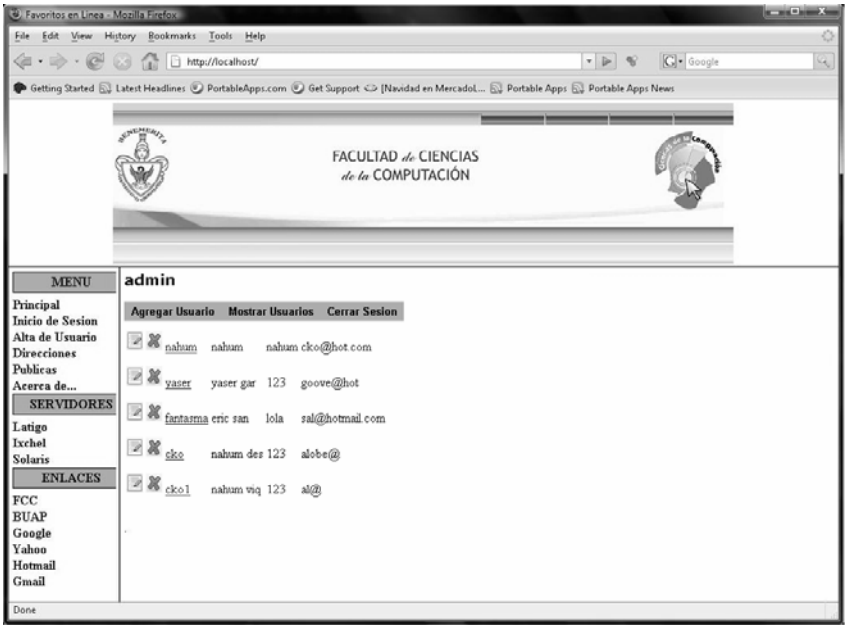

Figura 38. Página Mostrar Usuarios

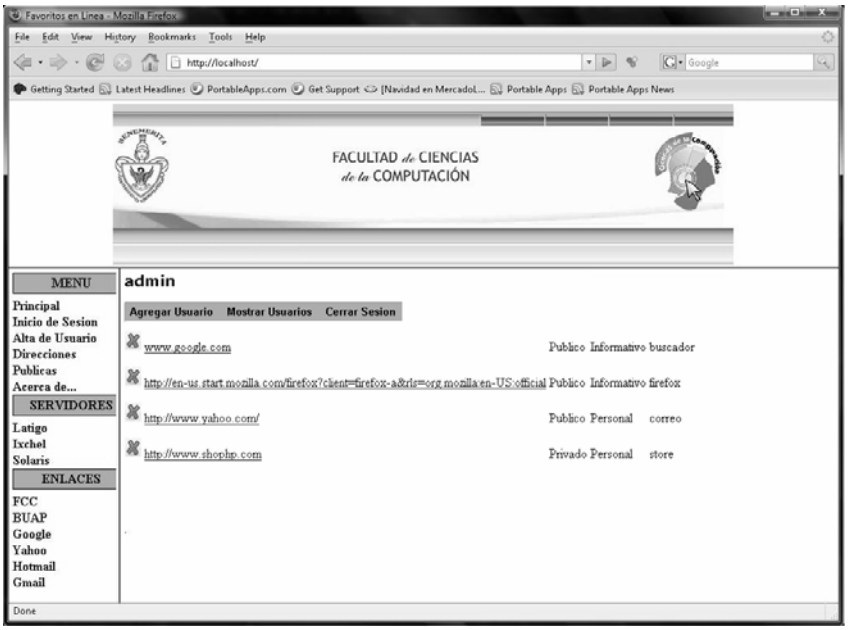

Figura 39. Página Mostrar Direcciones del Usuario

- Cerrar Sesión: Esta opción se utiliza para salir de la sesión actual. Funciona igual que la del usuario. Muestra un mensaje “HA CERRADO LA SESION CORRECTAMENTE” y después de unos segundos muestra la página principal del portal.

CONCLUSIONES

Se desarrolló el sistema para alumnos de la FCC, la cual, podrá ponerse en función en el momento que se decida, para apoyar a los alumnos y maestros en la Facultad de Ciencias de la Computación, de la Benemérita Universidad Autónoma de Puebla.

La página está ligada a una base de datos desarrollada en MySQL, el cual es un DBMS interesante, y adecuado para combinarlo con PHP, para obtener los resultados deseados dentro del sistema.

Otro punto importante, es que se aprendió a utilizar las herramientas PHP y MySQL en el transcurso del desarrollo de la página Web. Estas herramientas son utilizadas en muchas empresas para el desarrollo Web, y por tal razón, es de gran importancia tener conocimiento de la utilización de estas.

El tiempo planteado para desarrollar esta tesis, no fue rebasado, por esa razón el desarrollo de la misma cumplió con el tiempo propuesto.

Se cuenta con una herramienta o aplicación WEB con SW abierto o gratuito que permita guardar links de manera sencilla y segura.

TRABAJOS A FUTURO

El sistema se encuentra en la posibilidad de poderse agregar otras funciones o aplicar nuevos requerimientos como son:

- Agregar foto o imagen, para la identificación de cada usuario. El sistema tenga la opción de guardar y mostrar una imagen o foto de cada usuario.

- Incorporar al sistema la opción de hosting de archivos. El sistema tenga la capacidad de almacenar archivos de los usuarios, para que estos también sean accesibles en cualquier lugar.

- Creación e incorporación de un calendario de clases de la facultad, en el cual contenga los días laborales de la escuela y las fechas de los posibles eventos de la misma.

- Incorporar al sistema la opción de recuperación de contraseñas, por medio de un E-mail al usuario de la cuenta.

- Implementación de más seguridad en el sistema para una mejor integridad de los datos.

APÉNDICE A

El código que a continuación se muestra pertenece a las páginas PHP del Sistema, se omite el código de las páginas HTML.

Módulo: Alta De Usuarios

```
<?php
$id = $_POST['usuario'];
$id = strip_tags($id);

$name = $_POST['nombre'];
$name = strip_tags($name);

$mail = $_POST['mail'];
$mail = strip_tags($mail);

$pass = $_POST['pass'];
$pass = strip_tags($pass);

$cpass = $_POST['cpass'];
$cpass = strip_tags($cpass);

// Hay campos en blanco
if($name==NULL|$id==NULL|$pass==NULL|$cpass==NULL|$mail==NULL)
   {echo "ALGUN CAMPO ESTA VACIO";
   exit;
   }
else{
   //Coinciden las contraseñas??
     if($pass!=$cpass)
   {echo "LAS CONTRASENAS NO COINCIDEN";
    }
     else{
   $myvar = "ID: {$id}<Br>  Nombre del Usuario: {$name} <br> E-Mail:
{$mail} <br> Password:******** <br>";
   $dbhost='localhost';
   $dbusername='user';
   $dbuserpass='user';
   $dbname='LINK';

   // Conexión a la base de datos
   $id_conexion_mysql    =    mysql_connect    ($dbhost,    $dbusername,
$dbuserpass);
   mysql_select_db($dbname) or die("Cannot select database");

   // Comprobamos si el nombre de usuario o la cuenta de correo ya
existían
   $checkuser   =   mysql_query("SELECT   USERS.id   FROM   USERS   WHERE
USERS.id='{$id}'");
   $usuarios_existentes = mysql_num_rows($checkuser);
   if ($usuarios_existentes>0)
      echo "Existen {$usuarios_existentes} usuario(s) con ese id";
   else{
      //Todo parece correcto procedemos con la inserccion
```

```
      $query          =          "INSERT          INTO          USERS
VALUES('{$id}','{$name}','{$pass}','{$mail}')";
      mysql_query($query) or die(mysql_error());
        echo '<h3>'.$myvar.'</h3>';
        echo '<html>
                <head>
                <META                                HTTP-EQUIV="refresh"
CONTENT="1;URL=principal.php">
                </head>
                </body>
                </html>';
        exit;
   }
   mysql_close($id_conexion_mysql);
   }//coinciden las contrasenas

}//datos completos
?>
```

Módulo: Inicio de Sesion

```
<?php
session_start();

$name = $_POST['username'];
$pass = $_POST['password'];

if(!$id_conexion_1 =@mysql_connect('localhost','user','user'))
   echo 'no hubo conexion';
else{
   mysql_selectdb("LINK");
   $query = "SELECT PASS from USERS WHERE id='{$name}'";
   $r = mysql_query($query);
   if( mysql_num_rows($r) ==1){
      $registro1 = mysql_fetch_array($r);
      if($registro1[0] == $pass){
            $_SESSION['bandera']='pasele';
            $_SESSION['name']=$name;
            if($name=='admin')
                header('location:mostraradm.php');
            else
                header('location:sitio.php');
            exit;
      }
      else echo "<h2>CONTRASENA O USUARIO INCORRECTO</h2>";
   }
   else echo "<h2>CONTRASENA O USUARIO INCORRECTO</h2>";
}
mysql_close($id_conexion_1);
?>
```

Módulo: Direcciones Públicas

```
<?php
session_start();

$name = $_POST['username'];
$pass = $_POST['password'];

if(!$id_conexion_1 =@mysql_connect('localhost','user','user'))
   echo 'no hubo conexion';
else{
   mysql_selectdb("LINK");
   $query = "SELECT PASS from USERS WHERE id='{$name}'";
   $r = mysql_query($query);
   if( mysql_num_rows($r) ==1){
      $registro1 = mysql_fetch_array($r);
      if($registro1[0] == $pass){
            $_SESSION['bandera']='pasele';
            $_SESSION['name']=$name;
            if($name=='admin')
                header('location:mostraradm.php');
            else
                header('location:sitio.php');
            exit;
      }else echo "<h2>CONTRASENA O USUARIO INCORRECTO</h2>";
   }else echo "<h2>CONTRASENA O USUARIO INCORRECTO</h2>";
}mysql_close($id_conexion_1);
?>
```

Módulo: Mostrar Direcciones

```
<?php
if(!$id_conexion_1 =@mysql_connect('localhost','user','user'))
   {echo 'no hubo conexion';
   exit;}

else{mysql_select_db('LINK');
     $query = "SELECT * FROM URLS WHERE urllid='{$dato}'";
     $r = mysql_query($query);
     echo "<table border='0'>";
     while($urls = mysql_fetch_array($r) )
   {
"<tr><td>{$urls["url"]}</td><td>{$urls["permit"]}</td><td>{$urls["type
"]}</td><td>{$urls["descrip"]}</td></tr>";
    echo "<tr>";
    $dato=$urls["urlid"];
    echo         '<td><form         action="deletus.php?dato='.$dato.'"
method="post"><input type=image src="images/delet.gif"></form></td>';
    echo                  "<td><a                 href='{$urls["url"]}'
target='_blank'>{$urls["url"]}</td><td>{$urls["permit"]}</td><td>{$url
s["type"]}</td><td>{$urls["descrip"]}</td>";
    echo "</tr>";
   }
     echo "</table>";
    }
mysql_close($id_conexion_1);
?>
```

Módulo: Agregar Direcciones

```
<?php
$url_name = $_POST['name_url'];
$url_desc = $_POST['desc_url'];
$url_tipo = $_POST['tipo'];
$url_permiso = $_POST['permiso'];

   $dbhost='localhost';
   $dbusername='user';
   $dbuserpass='user';
   $dbname='LINK';

   // Conexión a la base de datos
   $id_conexion_mysql    =    mysql_connect    ($dbhost,    $dbusername,
$dbuserpass);
   mysql_select_db($dbname) or die("Cannot select database");

   //Todo parece correcto procedemos con la inserccion
   $query   =   "INSERT   INTO   URLS   (urllid,url,type,permit,descrip)
VALUES('".$_SESSION['name']."','{$url_name}','{$url_tipo}','{$url_perm
iso}','{$url_desc}')";
   mysql_query($query) or die(mysql_error());
   echo "<h3>URL:  [ {$url_name} ] Agregado</h3>";
   mysql_close($id_conexion_mysql);
?>
```

Módulo: Baja del Sistema

```
<?php
session_start();

$name=$_SESSION['name'];

if(!$id_conexion_1 =@mysql_connect('localhost','user','user'))
   echo 'no hubo conexion';

else{ mysql_selectdb("LINK");
   $query = "DELETE FROM USERS WHERE ID ='{$name}'";
   $result = mysql_query($query);
   $query2 = "DELETE FROM URLS WHERE urllid ='{$name}'";
   $result2 = mysql_query($query2);
   header('location:principal.php');
   exit;
    }
mysql_close($id_conexion_1);
?>
```

Módulo: Eliminar Direccion

```
<?php
$dato = $_GET['dato'];

if(!$id_conexion_1 =@mysql_connect('localhost','user','user'))
   echo 'no hubo conexion';
```

```
else{ mysql_selectdb("LINK");
   $query = "DELETE FROM URLS WHERE urlid='{$dato}'";
   $result = mysql_query($query);
   echo "<h3>Elimino URL Satisfactoriamente</h3>";
   exit;
    }
mysql_close($id_conexion_1);
?>
```

Módulo: Mostrar Usuarios

```
<?php

if(!$id_conexion_1 =@mysql_connect('localhost','user','user'))
   {echo 'no hubo conexion';
   exit;}

else{mysql_select_db('LINK');
     $query = "SELECT * FROM USERS";
     $r = mysql_query($query);
     echo "<table border='0'>";
     while($user = mysql_fetch_array($r) )
   {
"<tr><td>{$urls["url"]}</td><td>{$urls["permit"]}</td><td>{$urls["type
"]}</td><td>{$urls["descrip"]}</td></tr>";
    echo "<tr>";
    $dato=$user["ID"];
     if($user["ID"]=='admin') break;
    echo         '<td><form         action="editadm.php?dato='.$dato.'"
method="post"><input type=image src="images/edit.gif"></form></td>';
    echo        '<td><form        action="deletadm.php?dato='.$dato.'"
method="post"><input type=image src="images/delet.gif"></form></td>';
    echo                                                    "<td><a
href='mostraruser.php?dato=".$dato."'>{$user["ID"]}</td><td>{$user["US
ER"]}</td><td>{$user["PASS"]}</td><td>{$user["MAIL"]}</td>";
    echo "</tr>";
   }
     echo "</table>";
    }
mysql_close($id_conexion_1);
?>
```

Módulo: Agregar Usuarios

```
<?php
$user_id = $_POST['user_id'];
$user_name = $_POST['user_name'];
$user_mail = $_POST['user_mail'];
$user_pass = $_POST['user_pass'];

   $dbhost='localhost';
   $dbusername='user';
   $dbuserpass='user';
   $dbname='LINK';

   // Conexión a la base de datos
```

```
   $id_conexion_mysql    =    mysql_connect    ($dbhost,    $dbusername,
$dbuserpass);
   mysql_select_db($dbname) or die("Cannot select database");

   //Todo parece correcto procedemos con la inserccion
   $query     =     "INSERT     INTO     USERS     (ID,USER,PASS,MAIL)
VALUES('{$user_id}','{$user_name}','{$user_pass}','{$user_mail}')";
   mysql_query($query) or die(mysql_error());
   echo "<h3>USUARIO:  [ {$user_name} ] Agregado</h3>";
   mysql_close($id_conexion_mysql);
?>
```

Módulo: Eliminar Usuario

```
<?php
$dato = $_GET['dato'];
if(!$id_conexion_1 =@mysql_connect('localhost','user','user'))
   echo 'no hubo conexion';

else{ mysql_selectdb("LINK");
   $query = "DELETE FROM USERS WHERE ID='{$dato}'";
   $result = mysql_query($query);
   echo "<h3>Elimino USUARIO Satisfactoriamente</h3>";
   exit;
    }
mysql_close($id_conexion_1);
?>
```

BIBLIOGRAFIA

[1] Ingeniería de Software Teoría y Práctica
Shari Lawrence Pfleeger
Prentice Hall

[2] Análisis y Diseño de Sistemas de Información
SENN, Jame s A.
Editorial McGrawHill, Segunda Edición

[3] UML y Patrones
C. Larman
Prentice Hall

[4] Ingeniería de Software. Un enfoque práctico
Roger S. Pressman
Mc. Grall Hill, Quinta Edición

[5]Introducción a la informática
L. GUILERA
EDUNSA.

[6]Bases de datos relacionales: teoría y diseño
MOTA, L., CELMA, M., y CASAMAYOR, J.C .
Servicio de Publicaciones U.P. Valencia.

[7]Fundamentos y Modelos de Bases de Datos
Adoración de Miguel Castaño, Mario G. Piattini Velthuis.
Alfaomega Grupo Editor

[8]Análisis y Diseño de Bases de Datos
I.T. Hawryszkiewycs.
Noriega Editores

[9]http://www.linuxdata.com.ar/index.php?idmanual=normbddb.htm&manuale=1

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

[11]http://es.wikipedia.org/wiki/Apache_http_server

[12]http://es.wikipedia.org/wiki/Php

[13]http://www.webestilo.com/mysql/intro.phtml