# Benemérita
# Universidad Autónoma de Puebla

**FACULTAD DE CIENCIAS DE LA COMPUTACIÓN**



## "UNA PLATAFORMA DE APOYO AL APRENDIZAJE BASADA EN SERVICIOS WEB"

# TESIS

**QUE PARA OBTENER EL TITULO DE:**
**LICENCIADO EN CIENCIAS DE LA COMPUTACIÓN**


**PRESENTA:**
**FABIAN GARCÍA TOCHIHUITL**

**ASESOR:**
**DR. MARIO ANZURES GARCÍA**

**COASESOR:**
**DR. MARIO ROSSAINZ LÓPEZ**


**PUEBLA PUE. NOVIEMBRE 2010**

# Una Plataforma de Apoyo al Aprendizaje basada en Servicios Web

## ÍINDICE

## INDICE DE FIGURAS

## INDICE FIGURAS DEL APÉNDICES

## INDICE DE TABLAS

## Resumen

En los últimos años se ha incrementado de manera significativa el uso de herramientas web colaborativas en universidades, para apoyar el aprendizaje del alumno, de manera grupal, un ejemplo es el uso de MOODLE, Dokeos, etc.

La Facultad de Ciencias de la Computación (FCC) no cuenta con alguna plataforma colaborativa que apoye la interacción grupal para promover la habilidad de pensamiento crítico y establecer un ambiente de colaboración, es por ello, que en este trabajo de tesis se propone una plataforma colaborativa que proporciona un espacio de colaboración, comunicación y coordinación centrado en el aprendizaje de los alumnos de la FCC.

Dicha plataforma se denomina SISAAC (SIStema de Apoyo al Aprendizaje Colabroativo) y está basada en las metodologías del aprendizaje colaborativo o CSCL (para fomentar la construcción y retroalimentación del pensamiento crítico) y de las Redes Sociales (para crear vínculos en la que los alumnos puedan expresarse de manera libre y natural en temas de interés académico)

SISAAC se desarrollo bajo una arquitectura SOA (Arquitectura orientada a servicios), la cual es el soporte metodológico para los servicios web que son implementados para éste trabajo de tesis. Tal es el caso de los servicios web para validar a usuarios, para la creación de grupos, etc.

Para la implementación de los servicios web se adopto la plataforma .NET por la gama de herramientas que dicho *framework* provee al programador. El lenguaje de programación empleado es C# y Asp.Net para la interfaz grafica. Además se

Se utilizaron otras herramientas como JavaScript, CSS (*Cascading Style Sheet*) hojas de estilo en cascada y DOM (*Document Object Model*) para darle una mejor apariencia a la interfaz. Finalmente, la base de datos se genero a través del gestor de bases de datos SQL SERVER 2008.

SISAAC es una herramienta ideal para el Modelo Universitario Minerva (MUM) debido a que cumple con los requisitos que este modelo impone, tales como, uso de las Tecnologías de la Información y la Comunicación, fomentar el trabajo en grupo, las relaciones académicas entre alumnos, etc.

## 1. Introducción

En los últimos años, la web ha evolucionado de manera significativa, ha pasado de ser una web estática donde el usuario jugaba un papel pasivo, es decir, solo era observador a ser un web dinámica donde el usuario se vuelve el principal protagonista; es por ello, que la web ha generado un espacio colaborativo, que permite la colaboración, coordinación y comunicación entre personas, grupos y/o organizaciones tanto de tipo empresarial como educativo. Este nuevo espacio ha llevado a la creación de diferentes plataformas tanto asíncronas como síncronas, que se ejecutan en entornos heterogéneos. Tales como las plataformas de aprendizaje colaborativo o CSCL (*Computer Supported Collaborative Learning,* en español Aprendizaje Colaborativo Asistido por Computadora), que estudia cómo la tecnología computacional puede apoyar el proceso de aprendizaje realizado a través de la suma de los esfuerzos colaborativos de los estudiantes, trabajando en una tarea grupal. Ejemplos de CSCL son: MOODLE, Dokeos, Black Board, WebCT, eCollege, ATutor, Sakai Project y Claroline y las Redes Sociales como FaceBook, MySpace, Hi5, Tagged, Sonico, Metroflog y Fotolog.

Las plataformas *CSCL* favorecen la interacción social, promueven la habilidad de pensamiento crítico, establecen un ambiente de cooperación para plantear y modelar problemas, aumentan la autoestima de los estudiantes al reducir la ansiedad mediante la ayuda en grupo, construyen relaciones entre estudiantes desconocidos, desarrollan aptitudes positivas y constituyen un sistema de soporte social para los estudiantes. Por otra parte, las Redes sociales se enfocan en crear comunidades o grupos basados en el establecimiento de vínculos sociales, generados por medio de una serie de herramientas web muy sencillas de utilizar.

En general, este tipo de redes son muy útiles, ya que sirven como un espacio web de colaboración donde los usuarios se contactan y expresan, fomentando la comunicación y la colaboración. Sin embargo, las plataformas *CSCL* se restringen a mostrar el contenido de un curso y limitan a los estudiantes a adquirir el conocimiento que se ha determinado en este. Por su parte, las redes sociales permiten que los miembros de los grupos ahí formados se expresen libremente pero sólo con un objetivo social.

Es por ello, que este trabajo de tesis desarrollará una plataforma académica colaborativa, la cual estará fundamentada en las metodologías utilizadas en el aprendizaje colaborativo o *CSCL* y las Redes Sociales, y basado su desarrollo en la tecnología de servicios web. Esta plataforma proporcionará un espacio de trabajo compartido y de interacción enfocado al ámbito académico, en particular a la comunidad académica de la Facultad de Ciencias de la Computación (FCC); de tal manera que la plataforma permitirá y apoyará la comunicación, colaboración y retroalimentación en grupo. Por tanto, ofrecerá a estudiantes y/o profesores la oportunidad de organizase en grupos de estudio para intercambiar y/o adquirir información acerca de un tema o problema en común, lo que permite una mayor adquisición de conocimiento de un modo dinámico y natural para esa comunidad. Esta plataforma como una red académica permitirá aprender interactuando y

aprender compartiendo, ya que proporcionará un espacio académico que promoverá la interacción entre estudiantes y la compartición de información académica de su interés. Cada usuario puede integrarse a un grupo existente o crear un nuevo grupo de estudio al cual pueden integrarse nuevos estudiantes. Los grupos estarán relacionados con temas específicos de las áreas de la FCC, como son matemáticas, software y hardware.

La tesis se organiza de la siguiente manera: Capítulo II, presenta los fundamentos teóricos que sustentan la tesis, como son: la disciplina *groupware* en concreto CSCL, como SOA y servicios web, además de las herramientas que se involucran para la implementación de SISAAC. Capítulo III, se explica la metodología para el desarrollo de SISAAC (ciclo de vida del software, los diferentes modelos y herramientas UML), después se procede al análisis de requisitos y finalmente al diseño de SISAAC. Capítulo IV, detalla el proceso de desarrollo de SISAAC, la creación de los Servicios Web, scripts, hojas CSS que son utilizados en el desarrollo. Finalmente el Capítulo V, define la conclusión y las aportaciones del trabajo de tesis.

## 2. Marco teórico.

El presente capítulo tiene como objetivo mostrar los fundamentos teóricos que sustentan el trabajo de tesis, que se irán describiendo en secciones posteriores.

### 2.1. Aplicaciones Colaborativas.

Las aplicaciones colaborativas proporcionan un espacio de trabajo compartido, que proporciona la coordinación, comunicación e interacción entre un grupo de participantes que persiguen objetivos comunes, que pueden ser en ambientes educativos o de investigación. Estos espacios son o deben ser creados bajo un diseño adecuado a los fines instruccionales y bajo parámetros de administración y seguridad en el acceso, que garantice la autenticidad y privacidad de los objetos (archivos, diálogos, otros) de la zona de trabajo a sus miembros.
En la actualidad, un factor clave del éxito de las organizaciones (compañías e instituciones) es su capacidad para realizar el trabajo en grupo de manera adecuada y eficiente. Esto hace necesario considerar que, actualmente, hay una fuerte tendencia hacia el trabajo en grupos distribuidos, traspasando límites geográficos, organizacionales, culturales, etc. Debido a esto, se ha originado un interés, cada vez más creciente, por sistemas de “Trabajo Cooperativo Asistido por Computadora” (CSCW, *Computer Supported Cooperative Work*).

#### 2.1.1. Definición.

Podríamos definir CSCW como el campo de investigación que examina el diseño, adopción y utilización de tecnologías de la información cuyo objetivo principal es facilitar los procesos de interacción y colaboración en el trabajo cooperativo. El término fue acuñado en 1984 por Irene Greif [Sarin, 85] y Paul Cashman [Cashman, 84] en un intento de agrupar a investigadores de áreas que deseaban resolver un problema en común: el trabajo en grupo mediado por una Computadora.

Sin embargo, este campo no se refiere únicamente a la cooperación y trabajo en grupo, también tiene en cuenta las tecnologías empleadas en el desarrollo de las aplicaciones, la adopción del medio social y la representación. En general, envuelve cualquier diseño de software interesado en el comportamiento social y organizacional, incluyendo áreas como la ingeniería de comunicación, gestión de la información, sociología y las teorías organizacionales.

CSCW comenzó como un esfuerzo de los expertos en computación por aprender de economistas, psicólogos, sociólogos y de todos aquellos que pudiesen aportar algo para entender mejor las actividades de grupo; llegando a ser una área de investigación y desarrollo en computación en la que se comparten experiencias y discuten diferentes posibilidades y restricciones tecnológicas con respecto al trabajo en grupo. CSCW se orienta en tres componentes primarios: la tecnología que apoyará el proceso de colaboración, los tipos de usuarios que se benefician, y la importancia de relaciones de trabajo eficaces.

CSCW es un campo de investigación interdisciplinario que trata de entender la manera en que la gente trabaja en grupos utilizando las tecnologías de comunicación e información. CSCW investiga cómo el trabajo en grupo puede ser apoyado por tecnologías de la información y la comunicación para mejorar el rendimiento del grupo de personas involucradas en la ejecución de tareas comunes o interrelacionadas.

Podemos hacer mención de las Características de CSCW

- Comunicación entre miembros del grupo
- Compartir información.
- Coordinación y control de objetos compartidos.
- Compartir un espacio de trabajo, de organización y entendimiento común del proceso de trabajo.

GroupWare *(Group +Software*): Otro concepto importante es el de *Groupware*. El termino *Groupware* es utilizado para denotar aquellos productos o aplicaciones orientadas al soporte de trabajo en grupo. El termino lo establece Peter y Trudy Johnson-Lenz en 1981, vinculando directamente el trabajo en grupo *(Group)* con las herramientas software que lo soportan y facilitan (*Software tools*). De hecho CSCW también ha sido definido como "el estudio de las herramientas y técnicas de *groupware* así como sus efectos psicológicos, sociales y organizacionales" [Martínez 05].
Las aplicaciones *Groupware* (también conocidas como aplicaciones colaborativas) se definen como sistemas basados en computadoras que asisten a grupos de personas implicadas en una tarea (o meta), proporcionando una interfaz para un entorno compartido [Ellis 91]. Para algunos autores, los términos de CSCW y *Groupware* se refieren a lo mismo; sin embargo, la principal diferencia es que CSCW es la disciplina que analiza el trabajo en grupo mientras *Groupware* es la tecnología que lo permite [Martínez 05].

Tecnologías *Groupware* son aquellas herramientas que permiten el trabajo en grupo, mediante las cuales distintas personas situadas en diferentes localizaciones del mundo pueden trabajar en un mismo proyecto, comunicarse, cooperar, resolver problemas, competir o incluso negociar, para la elaboración de una tarea común. Normalmente, estas herramientas se utilizan vía Internet y pueden ser.

- Herramientas Síncronas: Son todas las herramientas en la que es necesario que todos los colaboradores estén conectados para llevar a cabo la colaboración (coincidencia temporal), por ejemplo los *Chat*. véase en la Tabla 2.1.
- Herramientas Asíncronas: Las cuales no necesitan que todos los colaboradores participen al mismo tiempo, sino que pueden ir contribuyendo al trabajo en diferentes plazos de tiempo (no coincidencia temporal) .Las más comunes: los Grupos de Noticias, Listas de distribución, Correo Electrónico, y los Foros de Discusión véase en la Tabla 2.1.

| | | INTERACCIÓN | |
|---|---|---|---|
| | | Coincidencia Temporal Sincronía | No Coincidencia Temporal Asíncrona |
| LOCALIZACION | Mismo Lugar (Local) | Actividades presénciales como presentaciones, Reuniones, etc. | Ordenadores Compartidos |
| | Diferente Lugar (remoto) | Actividades a tiempo real a través de video/audio conferencia, "Chat", navegación cooperativa, aplicaciones compartidas, pizarras Compartidas, etc. | Comunicación e intercambio de información independientemente del lugar y el momento en que se realicen mediante correo electrónico, conferencias electrónicas, noticias, lluvia De ideas, votaciones, etc. |

Tabla 2.1: Tabla con ejemplos de herramientas síncronas y asíncronas.

### 2.1.2. Características de las herramientas colaborativas

Principales características:

- Multiplataforma: Algunas herramientas son multiplataforma ya que utilizan estándares que pueden ser visualizados en cualquier ordenador, Mac, PC, Unix, etc.
- Utilizan un navegador: Los usuarios acceden a la información a través de navegadores existentes en el mercado (como Nestscape o Explorer).
- Estructura cliente/servidor. En el caso de las herramientas con la estructura cliente-servidor, no se requiere la instalación del software en el ordenador de usuario sino que éste se conecta al servidor que lo contiene.
- Permiten trabajar con documentos en diferentes formatos.
- Facilitan la interacción/comunicación.
- Acceso restringido: El acceso a los espacios de trabajo puede ser público o restringido, en tal caso los usuarios deben disponer de nombre de usuario y clave para acceder.
- Interfaz gráfica: utilizan una interfaz basada en la web para permitir la integración de diferentes elementos multimedia: texto, gráficos, vídeo, sonidos, animaciones, etc.
- Permiten almacenar, recuperar y modificar documentos con relativa facilidad.
- Algunas de las utilidades que presentan las herramientas para el trabajo colaborativo son:

- Aplicaciones compartidas. Permiten manipular un mismo documento simultáneamente por todos los miembros del grupo.
- Asignación de tareas. Facilita la asignación de tareas en trabajos determinados, para todos los integrantes del grupo.
- Bases de datos. Manejan la información compartida.
- Calendario. Herramienta que puede ser utilizada de forma individual o compartir una agenda con el grupo.
- *Chat.* Permiten la comunicación síncrona a través de texto.
- Convocatoria de reuniones. Facilita la convocatoria de reuniones, incluyendo el asunto de la reunión, quién la convoca y los asistentes.
- Lluvia de ideas. Permite al grupo generar una lista de ideas, cada miembro va añadiendo sus ideas al resto.
- Navegación compartida. Permite que los demás miembros del grupo puedan seguir un itinerario de navegación propuesto por otro miembro de forma simultánea.
- Notas. Permiten dejar notas breves a los otros miembros del grupo.
- Pizarra compartida. Posibilita la manipulación de un dibujo, gráfico o esquema por todo el grupo de forma simultánea.
- Video/audio conferencia. Permite una comunicación síncrona a través de video y/o sonido.
- Votaciones. Gestiona la votación de ideas por parte del grupo, normalmente con una respuesta afirmativa o negativa, exponiendo seguidamente los resultados de la votación.

### 2.1.3. CSCL.

El incremento en el uso de la computadora en la educación, y en particular la migración a cursos basados en la web en universidades, ha originado el interés entre los educadores en métodos no tradicionales para la enseñanza y aprendizaje especialmente en grupo. Se trata de un área de trabajo centrada en teorías de aprendizaje orientadas al aprendizaje colaborativo (CSCL) usando tecnologías informáticas.

CSCL se da en contextos de aprendizaje escolar donde el desarrollo personal y grupal se constituye en el vector de su funcionamiento, y donde la división de las labores no está predeterminada.

Los sistemas colaborativos tratan de encontrar un modelo que englobe los diferentes participantes, las tareas a realizar y los modos de colaboración, resaltando que el papel de la tecnología es facilitar la comunicación; así como también es gestor y organizador para dar soporte al trabajo de grupo en tareas de aprendizaje. Además, es conveniente proporcionar la posibilidad de registrar los procesos de trabajo, para que se puedan establecer modelos que permitan analizarlos, monitorizarlos y, si procede, intervenir para mejorarlos.

Las funcionalidades que pueden ofrecer los sistemas de soporte para aprendizaje colaborativo (CSCL) son variadas y entre otras se puede citar la mediación en el intercambio de información, el ofrecer mecanismos de ayuda en la toma de decisiones, facilitar la comunicación en relación a las tareas a realizar, u organizar y gestionar el conocimiento compartido que se genera a lo largo de la tareas

[Conklin 96]. Para los sistemas CSCL lo adecuado es desarrollar herramientas en las que se tengan en cuenta aspectos de psicología, pedagogía y tecnologías de información, para definir modelos computacionales a partir de los cuales se puedan derivar arquitecturas genéricas que permitan incorporar diferentes modelos de colaboración e intervención pedagógica para mejorar los procesos de enseñanza y aprendizaje en grupo. Por lo general, los ambientes CSCL involucran estudiantes distribuidos en grupos que trabajan juntos, sobre un problema o proyecto en común. En estos espacios colaborativos se presentan múltiples agentes que deben coordinar múltiples representaciones superficiales y conceptuales; hacen referencia a diferencias en sistemas conceptuales o modos de aproximación a un contenido en particular.
En consideración de lo anterior, definiremos en forma a priori el CSCL como una estrategia de enseñanza-aprendizaje por la cual interactúan dos o más sujetos para construir aprendizaje, a través de discusión, reflexión y toma de decisión, proceso en el cual los recursos informáticos actúan como mediadores.
CSCL proporciona beneficios en diversos ámbitos de los cuales hacemos mención de algunos de ellos.

- Beneficios Académicos:
  - Promover la destreza de pensamientos críticos del estudiante.
  - Envolver activamente al alumno en el proceso de aprendizaje.
  - Mejorar los resultados en el salón de clases.
  - Modelar por parte del estudiante las técnicas apropiadas para resolver problemas.
- Beneficios Sociales:
  - Desarrollar un sistema de soporte social para estudiantes.
  - Construir un entendimiento diverso entre estudiantes y maestros.
- Beneficios Psicológicos:
  - Incrementar la autoestima de los estudiantes.
  - Crear actitudes positivas hacia los maestros.

En la actualidad existen numerosas herramientas CSCL a continuación mencionamos las siguientes:

#### 2.1.3.1. MOODLE

MOODLE (*Objet-Oriented Dynamic Learning Environment* -Entorno de Aprendizaje Dinámico y Modular Orientado a Objetos) [Moodle, 2007] es un sistema para la gestión de cursos y sitios web basados en Internet de libre distribución. Que ayuda a los educadores a crear aprendizajes en línea. Este tipo de plataformas tecnológicas también se conocen como LMS (*Learning Management System*).

MOODLE fue creado por Martín Dougiamas, basó su diseño en las ideas del constructivismo en pedagogía que afirman que el conocimiento se construye en la mente del estudiante en lugar de ser transmitido a partir de libros o enseñanzas y sí en un ambiente de aprendizaje colaborativo. Un profesor que opera desde éste punto de vista crea un ambiente centrado en el estudiante que le ayuda a construir ese conocimiento con base en sus habilidades y conocimientos propios en lugar de

simplemente publicar y transmitir la información que se considera que los estudiantes deben conocer.

MOODLE tiene un Enfoque Pedagógico: La filosofía planteada de MOODLE incluye una aproximación constructiva y constructivista social de la educación, enfatizando que los estudiantes (y no solo los profesores) pueden contribuir a la experiencia educativa en muchas formas. MOODLE es suficientemente flexible para permitir una amplia gama de modos de enseñanza. Puede ser utilizado para generar contenido de manera básica o avanzada (por ejemplo páginas web) o evaluación.

Dentro de las Características generales de MOODLE encontramos:

- Promueve una pedagogía constructivista social (colaboración, actividades, reflexión crítica, etc.). Su arquitectura y herramientas son apropiadas para clases en línea, así como también como complementar el aprendizaje presencial. Tiene una interfaz de navegador de tecnología sencilla, y combatible.
- La instalación requiere una plataforma que soporte PHP y la disponibilidad de una base de datos. MOODLE tiene una capa de abstracción de bases de datos por lo que soporta los principales sistemas gestores de bases de datos.
- Se ha puesto énfasis en una seguridad sólida en toda la plataforma. Todos los formularios revisados, las cookies cifradas, etc. La mayoría de las áreas de introducción de texto (materiales, mensajes de foros, entradas de los diarios, etc.).
- MOODLE se distribuye gratuitamente como software libre (*open Source*) bajo la licencia Publica GNU. Básicamente esto significa que MOODLE tiene derechos de autor (*copyright)*. Puede copiar, usar y modificar MOODLE siempre que acepte: proporcionar el código fuente a otros, no modificar o eliminar la licencia original y los derechos de autor, y aplicar esta misma licencia a cualquier trabajo derivado de él.
- MOODLE puede funcionar en cualquier ordenador en el que pueda correr PHP y soporta varios tipos de bases de datos (en especial MySQL).

La palabra MOODLE como se menciono antes es el acrónimo de *Modular Objet-Oriented Dynamic Learning Environmen*t, lo que resulta fundamentalmente útil para programadores y teóricos de la educación. También es un verbo que describe el proceso de deambular perezosamente a través de algo, y hacer las cosas cuando se te ocurre hacerlas, una placentera chapuza que a menudo te lleva a la visión y la creatividad. Las dos acepciones se aplican a la manera en que se desarrolló MOODLE y a la manera en que un estudiante o profesor podría aproximarse al estudio o enseñanza de un curso en línea.

### 2.1.3.2 BlackBoard (WebCT)

Blackboard [Blackboard 05] es un sistema comercial de aprendizaje virtual, utilizado principalmente para instituciones educativas, cuenta con herramientas integradas que facilitan la creación de entornos de aprendizaje en la web. Proporciona diferentes tipos de herramientas: de presentación (diseño del curso), de comunicación, de elaboración de test, de auto evaluación, glosario, agenda, índice alfabético, búsquedas, conversaciones en vivo (chats), gestión y administración, etc.
El contenido del curso es la parte central del sistema y la posibilidad de comenzar la discusión entre el grupo de aprendizaje es provista como una característica suplementaria mediante herramientas colaborativas [Martínez 05]. Gracias a la flexibilidad de sus herramientas para el diseño de clases hace este entorno muy atractivo tanto para principiantes como usuarios experimentados en la creación de cursos en línea [Friss, 05].
La Tabla 2.2 muestra información detallada sobre los componentes de un curso en Blackboard.

| ÁREA | DESCRIPCIÓN |
|---|---|
| Anuncios | Los anuncios publican información oportuna y fundamental para el éxito del curso. |
| Información de personal | La información de personal proporciona contexto e información de contacto sobre profesores y asistentes. |
| Áreas del curso | Las áreas del curso pueden contener una amplia variedad de elementos de contenido, entre ellos: evaluaciones, actividades, unidades didácticas y archivos multimedia. |
| Comunicación | El área de comunicación permite a los usuarios:<br>• enviar y recibir mensajes,<br>• abrir tableros de discusión,<br>• entrar al aula virtual,<br>• ver listas,<br>• ver páginas de grupos. |
| Enlaces externos | Los enlaces externos conectan a los usuarios con el material educativo que está fuera del *Blackboard Academic Suite.* |
| Herramientas | Herramientas que se pueden usar en el curso: buzón de transferencia digital, editar página principal, información personal, calendario, ver calificaciones, manual del alumno, tareas, *The Electric Blackboard* y libreta de direcciones. |
| Mapa del curso | Permite navegar a través de un directorio en árbol contraíble. |

Tabla 2.2: Tabla con información detallada sobre los componentes de un curso *Blackboard*.

### 2.1.3.3. Dokeos

Dokeos [Dokeos 06] es un sistema de aprendizaje virtual basado en la Web, técnicamente conocido como un LMS (*Learning Management System*), CMC (*Course Management Sysem*) o VLE (*Virtual Learning Environment*).

Intuitivo y fácil de usar por parte de todos los usuarios (profesores, formadores, estudiantes, proveedores de formación continua, etc.), Dokeos ofrece una amplia gama de herramientas y facilita la creación y organización de contenidos interactivos y ejercicios.

Al margen de su facilidad de uso, Dokeos es un software de código libre, gratuito. El código de *Dokeos* está disponible para que cualquiera pueda hacer uso del mismo o para realizar adaptaciones que adapten el software a las necesidades específicas de un usuario.

Dokeos ofrece un eficiente y amigable entorno virtual que integra herramientas de creación de contenido, de creación de actividades, herramientas colaborativas, así como sofisticadas herramientas de seguimiento e informes sobre el desempeño de los alumnos en el curso (véase Figura 2.1).


Figura 2.1 Herramientas de Dokeos.

## 2.2. Cómputo Orientado a Servicios.

El cómputo orientado a servicios es un nuevo paradigma computacional que utiliza servicios como base fundamental para el desarrollo de aplicaciones de rápido y bajo costo y una fácil composición de aplicaciones distribuidas, incluso en entornos heterogéneos. La promesa del SOC *(Service Oriented Computing*) es un mundo de cooperación de servicios donde los componentes de una aplicación son ensamblados con un pequeño esfuerzo en una red de servicios que pueden ser débilmente acoplados para crear dinámicas de negocios flexibles y aplicaciones ágiles que pueden abarcar organizaciones y plataformas de computo.

En SOC se identifican los siguientes niveles de integración:

- La integración a nivel de Servicios Básicos *e-services:* servicios basados en Internet hacen énfasis en la integración de servicios con otras aplicaciones disponibles en Internet.
- Servicios compuestos *e-workflow:* está formado por servicios básicos, para construir un modelo basado en servicios; el cómputo orientado a servicios utiliza la arquitectura orientada a servicios (SOA), la cual es la metodología para reorganizar aplicaciones de software previamente desarrolladas, incorporándolas en un conjunto de servicios interconectados, cada uno accesible a través de interfaces y protocolos de mensajería.

### 2.2.1. SOA.

SOA (Service Oriented Architecture - Arquitectura Orientada a Servicios) permite la creación de sistemas altamente escalables que reflejan el negocio de la organización, a su vez brinda una forma estándar de exposición e invocación de servicios (comúnmente, pero no exclusivamente servicios web), lo cual facilita la interacción entre diferentes sistemas propios o de terceros, reduce los costos y el tiempo de desarrollo, es decir, los servicios pueden reutilizarse fácilmente y convertirse en nuevas aplicaciones compuestas, reduce costos de mantenimiento, los servicios reutilizables reducen el grado de complejidad interna del sistema, además permite aumentar la calidad de los servicios en múltiples ciclos de prueba de diferentes consumidores de servicios. SOA también permite reducir los costos de integración por lo que los servicios estandarizados pueden trabajar en conjunto, permitiendo que las aplicaciones dispares se conecten con rapidez y facilidad, reduciendo en general, el riesgo relacionado con el cumplimiento.

#### 2.2.1.1. Elementos de SOA.

Los elementos y las relaciones que generalmente intervienen en una Arquitectura Orientada a Servicios se muestran en la Figura 2.2 y son descritos a continuación:

- Servicio: es una funcionalidad diseñada, codificada y encapsulada de modo que sea realizable por otros servicios y aplicaciones, no es importante la tecnología en que este constituido.

- Proveedor de Servicios: se encarga de implementar un servicio y publicarlo en un Repositorio de Servicios.
- Repositorio de Servicios: En el repositorio se almacenan las descripciones de servicios, para que los servicios puedan ser solicitados por un Consumidor de Servicios.
- Consumidor de Servicios: Solicitará un servicio previamente publicado en el Repositorio de Servicios, obteniendo una descripción, que será utilizada para obtener los mensajes necesarios para invocar el servicio deseado.

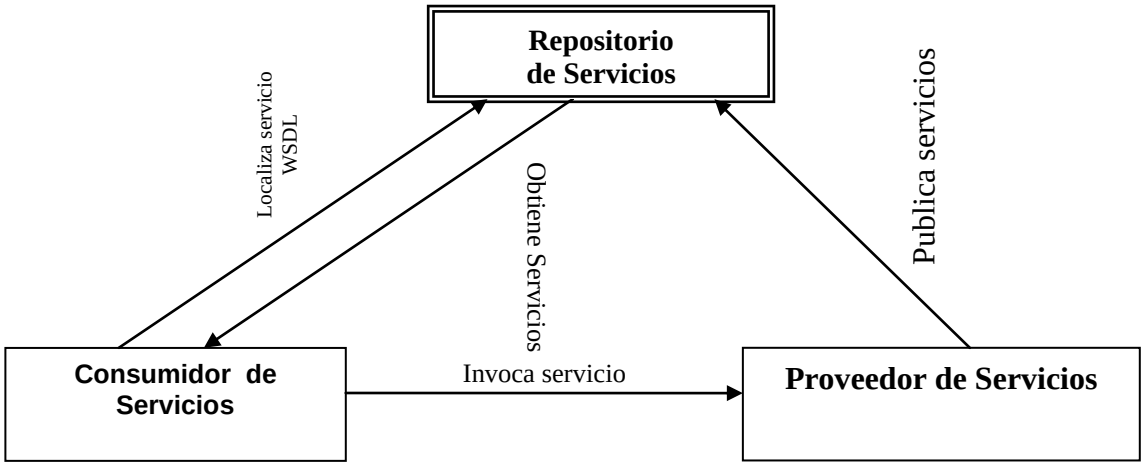



Figura 2.2 Arquitectura orientada a servicios básica

Una arquitectura Orientada a Servicios básica no abarca completamente todas las capacidades necesarias en la arquitectura de integración, para ello como complemento a esta arquitectura se han definido elementos que aportan nuevas funcionalidades (ver Figura 2.3) como: monitorización de procesos, servidores de reglas de negocios, y sobre todo motores de procesos, etc. Estos últimos han cobrado un auge con la aparición de nuevos estándares como BPEL, que permite orquestar la ejecución de diferentes servicios web de forma que se puedan implementar los procesos de negocios de una forma fácil y flexible.

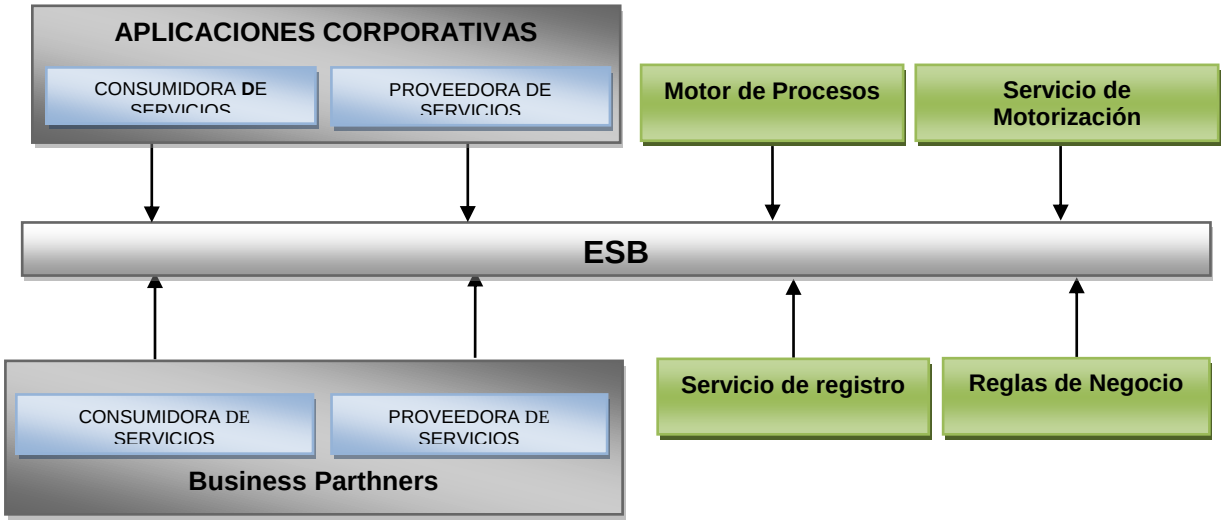



Figura 2.3 Arquitectura Orientada a Servicios extendida

### 2.2.2 Servicios Web.

Es un conjunto de estándares y protocolos que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes y ejecutados sobre cualquier plataforma, pueden utilizar los diferentes servicios web para intercambiar datos en redes de ordenadores como el internet, cuyo objetivo es ofrecer servicios; los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la web. La interoperabilidad (es la condición mediante la cual los sistemas heterogéneos pueden intercambiar procesos o datos) se consigue mediante la adopción de estándares abiertos, que son una especificación disponible públicamente para lograr una tarea específica.
El navegador Web es la puerta de entrada a todo un mundo de servicios ofrecidos por empresas y particulares. No obstante, las interacciones en la Web no son solo entre usuario y aplicación, sino también entre aplicaciones. Las empresas tienen que comunicarse con otras empresas para realizar transacciones más complejas, por ejemplo, al ser auditadas, al compartir recursos con sus filiales, etc. Para que las aplicaciones puedan resolver estas tareas se han creado estándares que se agrupan bajo la denominación de servicios web. Estos estándares permiten a las empresas categorizar el tipo de servicios que ofrecen, describir cómo deben interactuar con otras aplicaciones y definir como será codificada la información intercambiada entre un consumidor y el proveedor del servicio.
Los servicios Web serán los bloques básicos para la construcción de nuevas aplicaciones que puedan ser coordinadas u orquestadas. La orquestación de servicios web es un área de gran interés debido a que esta permite que diversas aplicaciones de negocios ofrecidas por distintos proveedores puedan comunicarse, acoplarse y conjuntarse en una aplicación orquestada, para obtener un mayor potencial en su funcionalidad.

Las características principales de los servicios web pueden ser resumidas en los siguientes puntos [OASIS, 02].

1. Son accesibles a través de la web. Gracias a la utilización de y la codificación de sus mensajes en SOAP (*Simple Object Access Protocol* - Protocolo Simple de Acceso a Objetos), protocolo estándar basado en XML (*Extensible Markup Language* -Lenguaje de Marcado Extensible).
2. Son interoperables. Debido a que los servicios pueden interactuar independientemente de la plataforma o lenguaje de programación en el cual hayan sido desarrollados.
3. Se describen a sí mismos. De esta forma una aplicación conoce cuál es la interfaz del servicio y podrá integrarlo y utilizarlo de manera automática. Dicha interfaz debe ser escrita en WSDL (*Web Services Description Language* - Lenguaje de Descripción de Servicios), que es un lenguaje basado en XML.
4. Son localizables. Mediante el uso de un repositorio de servicios Web, donde los servicios sean almacenados, publicados y descubiertos por distintas aplicaciones. Actualmente, UDDI *(Universal Description, Discovery*

*and Integration* - Descubrimiento e Integración Universal de la Descripción*)* representa la tecnología más utilizada.

### 2.2.2.1 Pila básica de Servicios Web

Los servicios Web están constituidos por estándares y a su vez pretenden ser un estándar mediante el cual sea posible construir sistemas a partir de piezas heterogéneas. Los principales estándares para el desarrollo de servicios web son:

#### 2.2.2.1.1 HTTP

**HTTP** (*Hypertext Transfer Protocol*, Protocolo de Transferencia de Hipertexto) [Microsoft], para la codificación y la transferencia de parámetros como pares de nombre y valor, junto con la semántica de solicitudes asociada. Cada uno se compone de una serie de encabezados de solicitud HTTP que, entre otras características, definen lo que el cliente solicita al servidor, el cual responde con una serie de encabezados de respuesta HTTP y los datos solicitados, si procede.

#### 2.2.2.1.2. XML

Es un metalenguaje extensible de etiquetas desarrollado por el *World Wide Web Consortium (*W3C), Principalmente para superar las limitaciones de expresividad en HTML.XML. Es un conjunto de reglas para definir etiquetas que permiten organizar un documento. XML es un metalenguaje que define la sintaxis utilizada para definir familias de lenguajes de etiquetas estructurados. XML no ha nacido sólo para su aplicación en Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y casi cualquier cosa imaginable.

1. Es un estándar abierto. XML es respaldado por muchas compañías con base tecnológica como IBM, Microsoft, Sun Microsystems, entre otros; quienes lo integran en sus aplicaciones y herramientas de desarrollo.
2. Simplicidad de sintaxis. XML representa cualquier tipo de datos y hace fácil su codificación y entendimiento.
3. Independencia del protocolo de transporte. XML no necesita de ningún protocolo de trasporte especial, solo necesita de un protocolo que pueda transferir texto o documentos simples.

Un documento XML está formado por el prólogo y por el cuerpo del documento.

- Prólogo: Aunque no es obligatorio, los documentos XML pueden empezar con unas líneas que describen la versión XML, el tipo de documento y otras cosas.
- El prólogo contiene:

- Una declaración XML. Es la sentencia que declara al documento como un documento XML.
- Una declaración de tipo de documento. Enlaza el documento con su DTD (definición de tipo de documento), o el DTD puede estar incluido en la propia declaración o ambas cosas al mismo tiempo.
- Uno o más comentarios e instrucciones de procesamiento.

- Cuerpo: A diferencia del prólogo, el cuerpo no es opcional en un documento XML, el cuerpo debe contener uno y solo un elemento raíz, característica indispensable también para que el documento esté bien formado.
- Elementos: Los elementos XML pueden tener contenido (más elementos, caracteres o ambos), o bien ser elementos vacíos.
- Atributos: Los elementos pueden tener atributos, que son una manera de incorporar características o propiedades a los elementos de un documento. Deben ir entre comillas.
- Entidades predefinidas: Entidades para representar caracteres especiales para que, de esta forma, no sean interpretadas como marcado en el procesador XML.
- Secciones CDATA: Es una construcción en XML para especificar datos utilizando cualquier carácter sin que se interprete como marcado XML. Solo se utiliza en los atributos. No confundir con 2(#PCDATA) que es para los elementos. Permite que caracteres especiales no rompan la estructura. Ejemplo:

  <![CDATA[ contenido especial: áéíóúñ& ]]>

- Comentarios: Comentarios a modo informativo para el programador que han de ser ignorados por el procesador. Los comentarios en XML tienen el siguiente formato:

  *<!--- Esto es un comentario --->*

**2.2.2.1.3. WSDL**

WSDL es un lenguaje XML usado para describir la interfaz de programación de un servicio Web [Christensen, 01]. Describe los requisitos del protocolo y los formatos de los mensajes necesarios para interactuar con los servicios listados en sus catálogos, las operaciones y mensajes que soporta se describen en abstracto y se ligan después al protocolo concreto de red y al formato del mensaje.

Esta descripción puede ser vista como un contrato entre el proveedor del servicio y el cliente, mediante el cual el proveedor del servicio indica: qué métodos se pueden invocar, qué tipos de datos utilizan esos métodos, qué protocolos de transporte utilizará para el envió y recepción de peticiones y como se accederá a los servicios.

La estructura de un documento WSDL se estructura de la siguiente forma:

- Definiciones < *definitions* >. Elemento raíz, el cual está constituido por los siguientes subelementos.
    - Tipos < t*ypes* >. Se definen los tipos de datos utilizados en el servicio web.
    - Mensajes < *message* >. Define los datos que participan en una operación. Cada mensaje puede tener una o varias partes y cada parte puede considerarse como si fueran los parámetros que se pasan en la llamada a una función en programación clásica o un método en programación Orientada a objetos.
    - Tipo de puertos < *portType* >. Define el servicio web, así como las operaciones posibles mediante dicho servicio y los mensajes vinculados. Cumple una función análoga a la de una biblioteca a la de una función de biblioteca en programación clásica o a la de una clase en programación Orientada a Objetos.
    - Enlaces < *binding* >. Se define el formato del mensaje y el protocolo para cada uno de los puertos.
    - Servicios <*Service* >. Especifica una colección de puertos por medio de los cuales es posible acceder al servicio. Cada puerto tendrá un URL asociado (*endpoint*) que indicará donde está ubicada la descripción el servicio.

**2.2.2.1.4. SOAP.**

SOAP es un protocolo estándar que define como dos objetos de diferentes procesos pueden comunicarse por medio de intercambio de datos XML,

Una vez definida la descripción de un servicio Web, se utiliza un protocolo de mensaje para poder solicitar alguna operación provista por el servicio.
SOAP codifica los mensajes de petición, respuesta, y errores a las peticiones a métodos de servicios web, enviándolos a través de la red [W3C, 03].

El mensaje SOAP (ver Figura 2.4) está compuesto por un *envelope* (sobre), cuya estructura está formada por los siguientes elementos: *header* (cabecera) y *body* (cuerpo).

- Sobre <*envelope*>: Representa el contenedor del mensaje. Elemento obligatorio utilizado para especificar cómo se codificará el mensaje. Los subelementos por los cuales está constituido son:

    - Cabecera <*header*>: Elemento opcional en el cual son definidos los actores que proveen el servicio.
    - Cuerpo <*Body*>: Elemento obligatorio en donde se especifican los métodos que serán solicitados con sus respectivos parámetros. Además es posible definir etiquetas de error (soap *faults*) para indicar si se produjeron errores al procesar peticiones.

- *Attachments* (opcional). Permite intercambiar o añadir datos que no sean XML al mensaje, por ejemplo imágenes.

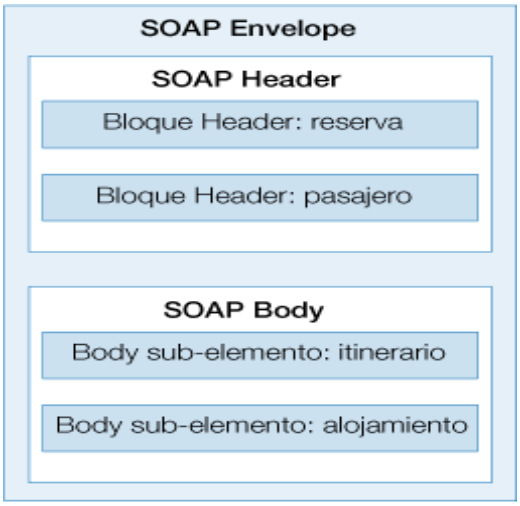



Figura 2.4. Esquema general del mensaje SOAP

**2.2.2.1.5. UDDI.**

Para que los desarrolladores puedan crear clientes que utilicen servicios es necesario que conozcan cuales son los servicios disponibles, qué es lo que ofrecen y cuáles son sus interfaces. La mejor opción consiste en utilizar internet para crear un registro estándar donde se pueda almacenar este tipo de información y acceder a ella. Esto es justamente lo que hace UDDI es un directorio de servicios distribuido el cual permite a los proveedores de servicios web dar a conocer sus ofertas de una forma estándar de tal manera que los clientes puedan consultarlos e invocarlos [OASIS, 02].

Un registro en el directorio UDDI es un archivo XML que describe una empresa y los servicios que ofrece. Un registro consta de tres partes:

- Páginas blancas: describen a la empresa que ofrece el servicio: nombre, dirección, contacto y otros identificadores conocidos.
- Páginas amarillas: categorización industrial basadas clasificaciones estándar como el *North American Industry Classification System y la Standard Industrial Classification.*
- Páginas verdes: información técnica sobre los servicios que aportan las propias empresas.

UDDI es uno de los estándares básicos de los servicios Web cuyo objetivo es ser accedido por los mensajes SOAP y dar paso a documentos WSDL, en los que se describen los requisitos del protocolo y los formatos del mensaje solicitado para interactuar con los servicios web del catálogo de registros.

UDDI es una pieza importante en el área de los servicios web. Sin él, las organizaciones que tienen servicios no podrían dar a conocer fácilmente a sus posibles clientes qué es lo que ofrecen, y los clientes no podrían conocer los datos que necesitan para acceder a esos servicios.

#### 2.2.2.2. Plataformas de Desarrollo Web.

En las siguientes secciones presentamos tres plataformas de desarrollo y sus características: .Net, NetBeans, PHP y NuSOAP.

##### 2.2.2.2.1. .NET.

La plataforma .NET de Microsoft es un componente de software que puede ser añadido al sistema operativo Windows. Provee un extenso conjunto de soluciones predefinidas para necesidades generales de la programación de aplicaciones, y administra la ejecución de los programas escritos específicamente con la plataforma. Esta solución es el producto principal en la oferta de Microsoft, y pretende ser utilizada por la mayoría de las aplicaciones creadas para la plataforma Windows.

Los principales componentes del marco de trabajo son:

- El conjunto de lenguajes de programación.
- La Biblioteca de Clases Base o BCL.
- El Entorno Común de Ejecución para Lenguajes o CLR por sus siglas en inglés.

Debido a la publicación de la norma para la infraestructura común de lenguajes (*CLI* por sus siglas en inglés), el desarrollo de lenguajes se facilita, por lo que el marco de trabajo .NET soporta ya más de 20 lenguajes de programación y es posible desarrollar cualquiera de los tipos de aplicaciones soportados en la plataforma con cualquiera de ellos, lo que elimina las diferencias que existían entre lo que era posible hacer con uno u otro lenguaje.

Algunos de los lenguajes desarrollados para el marco de trabajo .NET son: C#, Visual Basic, Delphi (Object Pascal), C++, J#, Perl, Python, Fortran, Prolog (existen al menos dos implementaciones, el P# y el Prolog.NET), Cobol y PowerBuilder.
La Biblioteca de Clases Base (BCL por sus siglas en inglés) maneja la mayoría de las operaciones básicas que se encuentran involucradas en el desarrollo de aplicaciones, incluyendo entre otras:

- Interacción con los dispositivos periféricos.
- Manejo de datos (ADO.NET).
- Administración de memoria.
- Cifrado de datos.
- Transmisión y recepción de datos por distintos medios (XML, TCP/IP).

- Administración de componentes Web que corren tanto en el servidor como en el cliente (ASP.NET).
- Manejo y administración de excepciones.
- Manejo del sistema de ventanas.
- Herramientas de despliegue de gráficos (GDI+).
- Herramientas de seguridad e integración con la seguridad del sistema operativo.
- Manejo de tipos de datos unificado.
- Interacción con otras aplicaciones.
- Manejo de cadenas de caracteres y expresiones regulares.
- Operaciones aritméticas.
- Manipulación de fechas, zonas horarias y periodos de tiempo.
- Manejo de arreglos de datos y colecciones.
- Manipulación de archivos de imágenes.
- Aleatoriedad.
- Generación de código.
- Manejo de idiomas.
- Auto descripción de código.
- Interacción con el API Win32 o Windows API.
- Compilación de código.

Esta funcionalidad se encuentra organizada por medio de espacios de nombres jerárquicos.
La Biblioteca de Clases Base se clasifica, en cuatro grupos clave:

- ASP.NET y servicios web XML.
- Windows Forms.
- ADO.NET.
- .NET.

#### 2.2.2.2.1.1. Microsoft Visual Studio.

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas operativos Windows. Soporta varios lenguajes de programación tales como Visual C++, Visual C#, Visual J#, ASP.NET y Visual Basic .NET, aunque actualmente se han desarrollado las extensiones necesarias para muchos otros.

Visual Studio permite a los desarrolladores crear aplicaciones, sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET (a partir de la versión net 2002). Así se pueden crear aplicaciones que se intercomuniquen entre estaciones de trabajo, páginas web y dispositivos móviles.

### 2.2.2.2.2. NetBeans.

NetBeans es un proyecto exitoso de código abierto con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios (¡y creciendo) en todo el mundo. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio 2000 y continúa siendo el patrocinador principal de los proyectos. Al día de hoy hay disponibles dos productos: el NetBeans IDE y NetBeans Platform.

NetBeans IDE es un entorno de desarrollo, una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el NetBeans IDE. NetBeans IDE es un producto libre y gratuito sin restricciones de uso.

También está disponible NetBeans Platform; una base modular y extensible usada como estructura de integración para crear grandes aplicaciones de escritorio. Empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones.

Ambos productos son de código abierto y gratuito para uso tanto comercial como no comercial. El código fuente está disponible para su reutilización de acuerdo con la *Common Development and Distribution License* (CDDL) v1.0 *and the GNU General Public License* (GPL) v2.

NetBeans IDE es mucho más que un Java IDE: NetBeans IDE es una herramienta de desarrollo modular para una amplia gama de tareas de desarrollo. La base de IDE incluye un editor avanzado en varios idiomas, depurador y perfilador de la integración, el control de versiones de archivos, funciones de colaboración y único desarrollador.

Ctrl +i acceso directo para realizar una búsqueda sensible al contexto para el IDE, los conjuntos de ayuda, y todos los proyectos abiertos. Va a encontrar el término de búsqueda no sólo en los archivos, tipos o símbolos, sino también todas las acciones relacionadas con el menú, los paneles de opción, y la documentación.

Enseguida se hace una descripción de algunas características de NetBeans IDE:

- Instalador personalizado: Seleccione la descarga NetBeans IDE que proporciona las características que usted necesita. Puede elegir la descarga con todas las características y luego configurar la instalación de manera que sólo las funciones que necesita realmente se instalen, o que elija una de las descargas más pequeña que tiene sólo un subconjunto de las funciones.
- Administrador de Plugin: Siempre se puede usar el Administrador de complementos en el menú Herramientas para agregar, quitar o actualizar

conjuntos de características de Java SE, Java EE, Java ME, JavaFX, Ruby, Groovy, PHP, C / C + +, UML, o el desarrollo de SOA, incluyendo una amplia variedad de otras características de los proveedores de terceros.

- Personaliza tu Proyectos: Configurar propiedades y dependencias de compilación entre sus proyectos, y utilizar rutas relativas biblioteca para que los proyectos compartibles. Tienda configuraciones de proyectos para diversas fases de desarrollo diferentes. El IDE se basa ejecutables binarios y archivos comprimidos en varios formatos, incluidos los JAR, WAR, EAR, NBM, JNLP Web Start, y los archivos ZIP, listos para su distribución.
- Platillas y aplicaciones de ejemplo: Puede crear un proyecto de forma libre sobre la base de un script de construcción existente y fuentes, o iniciar proyectos de una plantilla. El IDE incluye plantillas y ejemplos de proyectos para las aplicaciones Java SE, aplicaciones móviles, aplicaciones web, aplicaciones empresariales, las aplicaciones de JavaFX, NetBeans plugins, Groovy, PHP, C y C + +, Ruby y Ruby on Rails.
- Bases de Datos y servicios: La ventana de servicios le da acceso a todas sus bases de datos y servidores web, así como los servicios web y Enterprise JavaBeans (EJB). Usted puede iniciar y parar los servidores, y arrastrar y soltar los servicios web y EJB en sus proyectos.
- Ventanas deslizables: El área de trabajo del IDE es totalmente personalizable: Las fichas de personalizar las acciones en la barra de herramientas y arrastrar y colocar en la ventana IDE para adaptarse a su flujo de trabajo individual.
- Mover cualquier elemento del editor arrastrando y soltar fuera del IDE: se convertirá en una ventana independiente que se puede pasar a una segunda pantalla. Para integrar el elemento en la ventana.
- El editor soporta varios idiomas, incluyendo Java, Ruby, C / C + +, XML, HTML, RHTML, PHP, Groovy, Javadoc, JavaScript y JSP. Puede ser extendida para soportar cualquier otro idioma.
- Soporte para servicios web.

#### 2.2.2.2.3 PHP+NuSOAP

PHP es el acrónimo de *Hipertext Preprocesor*. Es un lenguaje de programación del lado del servidor gratuito e independiente de plataforma, rápido, con una gran librería de funciones y mucha documentación.

Un lenguaje del lado del servidor es aquel que se ejecuta en el servidor web, justo antes de que se envíe la página a través de Internet al cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red, y otras tareas para crear la página final que verá el cliente. El cliente solamente recibe una página con el código HTML resultante de la ejecución de la PHP. Como la página resultante contiene únicamente código HTML, es compatible con todos los navegadores.

Fue creado originalmente en 1994 por Rasmus Lerdorf, pero como PHP está desarrollado en política de código abierto, a lo largo de su historia ha tenido muchas contribuciones de otros desarrolladores.
Este lenguaje de programación está preparado para realizar muchos tipos de aplicaciones web gracias a la extensa librería de funciones con la que está dotado. La librería de funciones cubre desde cálculos matemáticos complejos hasta tratamiento de conexiones de red, por poner dos ejemplos.

Algunas de las más importantes capacidades de PHP son: compatibilidad con las bases de datos más comunes, como MySQL, mSQL, Oracle, Informix, y ODBC, por ejemplo. Incluye funciones para el envío de correo electrónico, cargar archivos, crear dinámicamente en el servidor imágenes en formato GIF, incluso animadas y una lista interminable de utilidades adicionales.

NuSOAP es una herramienta de código abierto que permite el envío de mensajes SOAP sobre el protocolo HTTP, lo cual facilita la creación de servicios web bajo PHP. Actualmente PHP no incluye soporte para el protocolo SOAP, por lo que hay que utilizar herramientas externas al lenguaje para facilitar el desarrollo de los servicios web.

La instalación de NuSOAP es muy sencilla, simplemente hay que acceder a la página http://sourceforge.net/progects/nusoap/ y descargar el fichero comprimido que contiene las clases, descomprimirlo en algún directorio del servidor y está listo para programar servicios web.

### 2.3. Bases de Datos.

Las Bases de Datos [Silberschatz, *02*] se han constituido como una de las herramientas más ampliamente difundidas en la actualidad en cuanto al almacenamiento y recuperación de Información en todos los campos a nivel científico, social, económico, político y cultural.

#### 2.3.1. Definiciones.

- Una Base de Datos es una colección de datos que contiene información relevante para una empresa.
- Un Sistema gestor de bases de datos (SGBD) consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a dichos datos. El objetivo principal de un SGBD es proporcional una forma de almacenar y recuperar la información de una base de datos de manera que sea tanto práctica como eficiente.

  Los sistemas de bases de datos se diseñan para gestionar grandes cantidades de información. La gestión de los datos implica tanto la definición de estructuras para almacenar la información como la provisión de mecanismos para la manipulación de la información. Además, los sistemas de bases de datos deben proporcionar la fiabilidad de la información

almacenada, a pesar de las caídas del sistema o los intentos de acceso sin autorización. Si los datos van a ser compartidos entre diversos usuarios, el sistema debe evitar posibles resultados anómalos.
Dado que la información es tan importante en la mayoría de las organizaciones, los científicos informáticos han desarrollado un amplio conjunto de conceptos y técnicas para la gestión de los datos.

### 2.3.2. Modelo Entidad Relación.

El modelo Entidad Relación (E-R) es un modelo de datos de alto nivel. Está basado en una percepción de un mundo real que consiste en una colección de objetos básicos, denominados entidades, y de relaciones entre estos objetos. Se desarrolló para facilitar el diseño de bases de datos permitiendo la especificación de un esquema de la empresa que representa la estructura lógica completa de una base de datos. El modelo de datos E-R es uno de los diferentes modelos de datos semánticos; el aspecto semántico del modelo yace en la representación del significado de los datos. El modelo E-R es extremadamente útil para hacer corresponder los significados e interacciones de las empresas del mundo real con un esquema conceptual. Debido a esta utilidad, muchas herramientas de diseño de bases de datos se basan en los conceptos del modelo E-R.

#### 2.3.2.1. Conjunto de Entidades.

Una entidad es una «cosa» u «objeto» en el mundo real que es distinguible de todos los demás objetos. Por ejemplo, cada persona en un desarrollo es una entidad. Una entidad tiene un conjunto de propiedades, y los valores para algún conjunto de propiedades pueden identificar una entidad de forma unívoca. Una entidad puede ser concreta, como una persona o un libro, o puede ser abstracta, como un préstamo, unas vacaciones o un concepto.

Un conjunto de entidades es un conjunto de entidades del mismo tipo que comparten las mismas propiedades, o atributos. Las entidades individuales que constituyen un conjunto se llaman la extensión del conjunto de entidades.

Un conjunto de entidades puede no tener suficientes atributos para formar una clave primaria. Tal conjunto de entidades se denomina conjunto de entidades débiles.
Un conjunto de entidades que tiene una clave primaria se denomina conjunto de entidades fuertes.
Una entidad se representa mediante un conjunto de atributos. Los atributos describen propiedades que posee cada miembro de un conjunto de entidades. La designación de un atributo para un conjunto de entidades expresa que la base de datos almacena información similar concerniente a cada entidad del conjunto de entidades; sin embargo, cada entidad puede tener su propio valor para cada atributo.

Una base de datos incluye así una colección de conjuntos de entidades, cada una de las cuales contiene un número de entidades del mismo tipo.

Formalmente, un atributo de un conjunto de entidades es una función que asigna al conjunto de entidades un dominio. Como un conjunto de entidades puede tener diferentes atributos, cada entidad se puede describir como un conjunto de pares (atributo, valor), un par para cada atributo del conjunto de entidades.

Los valores de los atributos que describen una entidad constituirán una porción significante de los datos almacenados en la base de datos.

Un atributo, como se usa en el modelo E-R, se puede caracterizar por los siguientes tipos de atributo.

- Atributos simples y compuestos**:** los atributos simples no están divididos en subpartes, Los atributos compuestos, en cambio, se pueden dividir en subpartes (es decir, en otros atributos). Por ejemplo, nombre-cliente podría estar estructurado como un atributo compuesto consistente en nombre, primer-apellido y segundo-apellido. Usar atributos compuestos en un esquema de diseño es una buena elección si el usuario desea referirse a un atributo completo en algunas ocasiones y, en otras, a algún componente del atributo.
  Los atributos compuestos ayudan a agrupar los atributos relacionados, haciendo los modelos más claros.
- Atributos monovalorados y multivalorados: Los atributos que solo pueden tener un único valor se denominan monovalorados. Los que pueden tener simultáneamente más de un valor se denominan multivalorados.
- Atributos derivados. Se denomina atributo derivado a aquel cuyo valor puede ser obtenido de los valores de otros atributos o entidades relacionadas. Por ejemplo, el atributo nombre completo podría obtenerse de loa atributos nombre, primer apellido y segundo apellido. El valor de un atributo derivado normalmente no se almacena, sino que se calcula cuando sea necesario.
  Un atributo toma el valor nulo (NULL) cuando no tiene ningún valor, o cuando el valor del atributo se desconoce, ya sea porque está perdido (El valor existe pero se ignora cuál es) o desconocido (no se sabe si el valor existe o no). El valor nulo también puede indicar “no aplicable”.

### 2.3.2.2. Conjunto de relaciones.

- Una relación es una asociación entre diferentes entidades.
- Un conjunto de relaciones es un conjunto de relaciones del mismo tipo. Formalmente es una relación matemática con $n$ > = 2 de conjuntos de entidades (posiblemente no distintos). Si $E_1$, $E_2$,…,$E_n$ son conjuntos de entidades, entonces un conjunto de relaciones R es un subconjunto de: {($e_1$, $e_2$,…, $e_n$) | $e_1 \in E_1$, $e_2 \in E_2$,…, $e_n \in E_n$} donde ($e_1$, $e_2$,… $e_n$) es una relación.

- La asociación entre conjuntos de entidades se conoce como *participación*; es decir, los conjuntos de entidades $E_1$, $E_2$,…, $E_n$ participan en el conjunto de relaciones R. Un ejemplar de relación en un esquema E-R representa que existe una asociación entre las entidades denominadas en la empresa del mundo real que se modela.
- La función que realiza una entidad en una relación se llama papel de la entidad. Los papeles son útiles cuando se necesita aclarar el significado de una relación. Un ejemplo podría ser cuando una entidad participa más de una vez en una relación (conjunto de relaciones recursivo). En este caso es más claro hacer explícitos los papeles.
- El número de conjuntos de entidades que participan en un conjunto de relaciones se denomina grado. La mayor parte de las relaciones en las bases de datos son binarias, es decir, de grado 2, pero ocasionalmente pueden aparecer conjuntos de relaciones en las cuales participan 3 o más conjuntos de entidades.

#### 2.3.2.3. Restricciones.

Un modelo E-R puede definir ciertas restricciones que los contenidos de la base de datos deben acatar.

##### 2.3.2.3.1. Correspondencia de cardinalidades.

La correspondencia de cardinalidades, o razón de cardinalidad, expresa el número de entidades a las que otra entidad puede estar asociada vía un conjunto de relaciones.
Para un conjunto de relaciones binarias R entre los conjuntos de entidades A y B, la correspondencia de cardinalidades debe ser una de las siguientes:

- Uno a uno. Una entidad en A se asocia con a lo sumo una entidad en *B*, y una entidad en *B* se asocia con *a lo sumo* una entidad en *A.*
- Uno a varios. Una entidad en *A* se asocia con cualquier número de entidades en *B* (ninguna o varias) Una entidad en *B*, sin embargo, se puede asociar con *a lo sumo* una entidad en *A.*
- Varios a uno. Una entidad en *A* se asocia con *a lo sumo* una entidad en *B*. Una entidad en *B*, sin embargo, se puede asociar con cualquier número de entidades (ninguna o varias) en *A.*
- Varios a varios. Una entidad en *A* se asocia con cualquier número de entidades (ninguna o varias) en *B*, y una entidad en *B* se asocia con cualquier número de entidades (ninguna o varias) en *A.*

##### 2.3.2.3.2. Restricciones de participación.

La participación de un conjunto de entidades *E* en un conjunto de relaciones *R* se dice que es total si cada entidad en *E* participa al menos en una relación en R. Si sólo algunas entidades en *E* participan en relaciones en *R*, la participación del conjunto de entidades *E* en la relación *R* se llama parcial.

### 2.3.2.4. Claves

Es necesario tener una forma de identificar unívocamente a la entidad. Para ello utilizaremos uno o varios atributos para formar una clave que nos permite distinguir las entidades entre sí. Dos entidades diferentes tendrán, por tanto, claves diferentes.

#### 2.3.2.4.1. Conjuntos de entidades

Una superclave es un conjunto de uno o más atributos que, tomados colectivamente, permiten identificar de forma única una entidad en el conjunto de entidades.
El concepto de una superclave no es suficiente para lo que aquí se propone, ya que, como se ha visto, una superclave puede contener atributos innecesarios. Tal y como se ha definido, si K es una superclave, también lo es cualquier superconjunto de K. A menudo interesan las superclaves tales que los subconjuntos propios de ellas no son superclave. Tales superclaves mínimas se llaman claves candidatas.
Pueden existir distintas claves candidatas. Se denominará clave primaria a aquella clave candidata que es elegida por el diseñador de la base de datos como identificar principal.
La clave primaria se debería elegir de manera que sus atributos nunca, o muy raramente, cambien.

#### 2.3.2.4.2. Conjuntos de relaciones.

De igual manera que se definió una clave primaria para las entidades que permite distinguir entre las diferentes entidades del conjunto. Se necesita un mecanismo similar para las relaciones.

Sea *R* un conjunto de relaciones que involucra los conjuntos de entidades $E_1$, $E_2$,…, $E_n$. Sea *clave-primaria* ($E_i$) el conjunto de atributos que forma la clave primaria para el conjunto de entidades $E_i$.

Asúmase por el momento que los nombres de los atributos de todas las claves primarias son únicos y que
Cada conjunto de entidades participa sólo una vez en la relación.

Si el conjunto de relaciones *R* no tiene atributos asociados, entonces el conjunto de atributos:

*clave-primaria(E*1*)* ∪ *clave-primaria(E*2*)* ∪ … ∪ *clave-primaria(En)*

Identifica de forma univoca a una relación individual en el conjunto R.

Dicho de otra forma, la combinación de las claves primarias de las entidades relacionadas forman la clave de la relación.
Si el conjunto de relaciones *R* tiene atributos $a_1,a_2,\ldots,a_m$ asociados a él, entonces el conjunto de atributos

$$clave\text{-}primaria(E_1) \cup clave\text{-}primaria(E_2) \cup \ldots \cup clave\text{-}primaria(E_n) \cup \{a_1, a_2,\ldots,a_m\}$$

Describe una relación individual en el conjunto R. En ambos casos, la clave primaria será la misma.

$$clave\text{-}primaria(E_1) \cup clave\text{-}primaria(E_2) \cup \ldots \cup clave\text{-}primaria(E_n)$$

En el caso de que los nombres de atributos de las claves primarias no sean únicos en todos los conjuntos de entidades, los atributos se renombran para distinguirlos. Si un conjunto de entidades participe más de una vez en un conjunto de entidades se usa el nombre de papel en lugar del nombre del conjunto de entidades para formar un nombre único de atributos.

### 2.3.2.5. Diagrama Entidad - Relación.

La estructura lógica general de base de datos se puede expresar gráficamente mediante un diagrama E-R. Los diagramas son simples y claros, cualidades que pueden ser responsables del amplio uso del Modelo E-R. Tal diagrama consta de los siguientes componentes principales:

- Rectángulos, que representan conjuntos de entidades.
- Elipses, que representan atributos.
- Rombos, que representan relaciones.
- Líneas, que unen atributos a conjuntos de entidades y conjuntos relaciones de entidades a conjuntos relaciones.
- Elipses dobles, que representan atributos multivaluados.
- Líneas dobles, que indican participación total de una entidad en un conjunto de relaciones.
- Rectángulos dobles, que representan conjuntos de entidades débiles.

## 2.3.3 Modelo Relacional.

El modelo relaciona es un modelo de menor nivel. Usa una colección de tablas para representar tanto los datos como las relaciones entre los datos. Su simplicidad conceptual ha conducido a su adopción general; actualmente, una vasta mayoría de productos de bases de datos se basan en el modelo relacional. Los diseñadores formulan generalmente el diseño del esquema de la base de datos modelando primero los datos en alto nivel, usando el modelo E-R, y después

traduciéndolo al modelo relacional. Ha conseguido la posición principal debido a su simplicidad, que facilita el trabajo del programador.

### 2.3.3.1. Estructura de las Bases de Datos Relacionales.

Una base de datos relacional consiste en un conjunto de tablas, a cada una de las cuales se le asigna un nombre exclusivo

#### 2.3.3.1.1 Estructura Básica:

Siguiendo la terminología del modelo relacional se puede hacer referencia a los elementos que componen la estructura básica de las bases de datos relacionales, los cuales describen a continuación:

1) Relación [Avda, 04]. Es la estructura básica del modelo relacional. Con una relación es posible representar tanto instancias de una entidad del universo real como interrelaciones entre entidades de distinto tipo. Es capaz de recoger interrelaciones de cardinalidad múltiple. Su representación informal es una tabla.
2) Dominio. Es el conjunto válido de valores de referencia para definir propiedades o atributos. Un dominio es un conjunto nominado y
3) homogéneo de valores Véase en la Figura 2.5.
   Cada dominio puede definirse de dos maneras:

- Por extensión (enumeración de sus elementos): Días de la semana = {lunes, martes, miércoles, jueves, viernes, sábado, domingo}.
- Por intensión (mediante una propiedad que recoja el recorrido de sus valores admisibles): Edad:entero/0<= edad<= 200.

Un dominio compuesto se puede definir como una combinación de dominios simples a la que se puede aplicar ciertas restricciones.
Ej.: el dominio compuesto denominado Fecha se construye por agregación de los dominios simples Día, Mes y Año, incorporando las restricciones a fin de que no aparezcan valores inválidos como: 29/2/2003, 31/4/2004.

4) Tupla : Es una ocurrencia o instancia dentro de una relación. Una tupla permite referenciar una instancia de una entidad en el universo o la interrelación específica o concreta entre instancias de entidades. Su representación informal es una fila. Una relación tiene un “conjunto” de tuplas. La relación es el elemento fundamental del modelo relacional. Véase en la Figura 2.6

- Atributo: Representa las propiedades de la relación. Un atributo, necesariamente ha de definirse sobre un dominio. Su representación informal es una columna. Un atributo (A) es la interpretación de un determinado dominio en una relación, es decir el “papel” que juega el dominio en la misma.

$$D = Dom\ (A) \Rightarrow D \text{ es el dominio de } A$$

- Un atributo está siempre asociado a una relación, mientras que un dominio tiene existencia propia con independencia de las relaciones que existan en el modelo.
- Un atributo representa una propiedad de una relación.
- Un atributo toma valores de un dominio.
- Varios atributos distintos (de la misma o de diferentes relaciones) pueden tomar sus valores del mismo dominio.

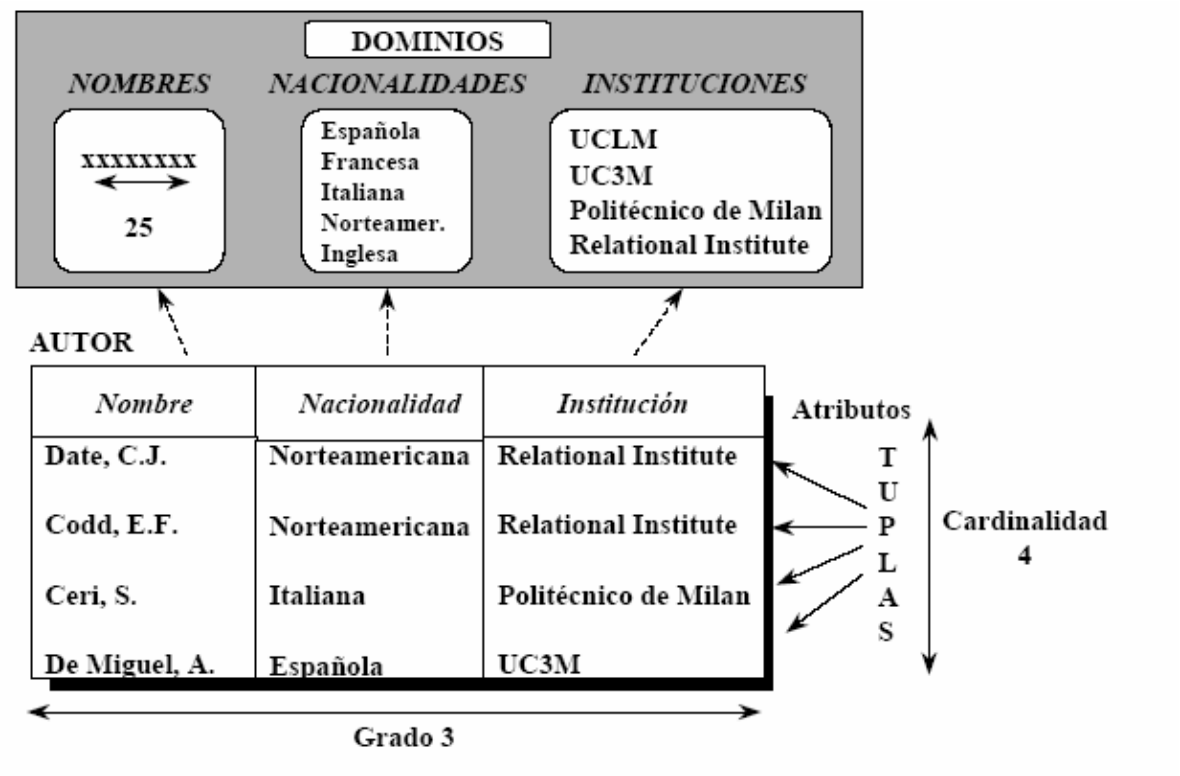



*Figura 2.5: Representación de la relación autor*

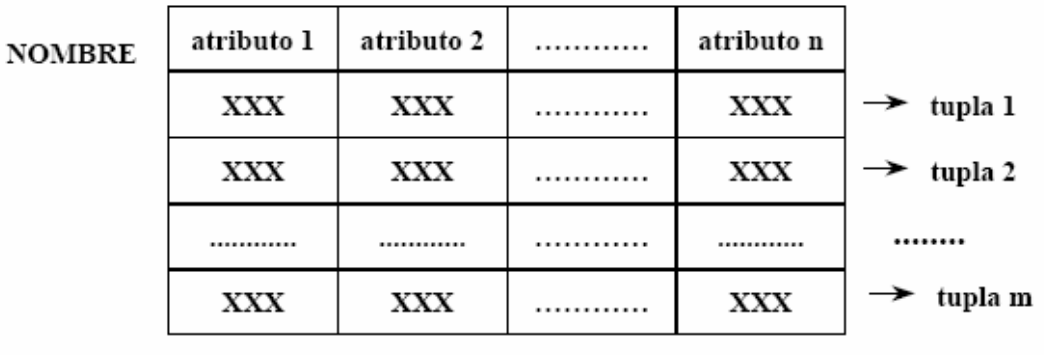



Figura 2.6: Representación de la relación en forma de tabla

Grado: Se define como el número de dominios $D_i$

Cardinalidad: Se define como el número de tuplas de la relación.

5) Claves: Sea *R* el esquema de una relación. Si se dice que un subconjunto *K* de *R* es una *superclave* de *R* para las relaciones *r*(*R*) en las que no hay dos tuplas diferentes que tengan los mismos valores en todos los atributos de *K*. Es decir, si $t_1$ y $t_2$ están en *R* y $t_1 \neq t_2$, entonces $t_1[K] \neq t_2[K]$.

   Si el esquema de una base de datos relacional se basa en las tablas derivadas de un esquema E-R es posible determinar la clave primaria del esquema de una relación a partir de las claves primarias de los conjuntos de entidades o de relaciones de los que se deriva el esquema:

- Conjunto de entidades fuertes. La clave primaria del conjunto de entidades se convierte en la clave primaria de la relación.
- Conjunto de entidades débiles. La tabla y, por tanto, la relación correspondientes a un conjunto de entidades débiles incluyen:
    - Los atributos del conjunto de entidades débiles.
    - La clave primaria del conjunto de entidades fuertes del que depende el conjunto de entidades débiles.

La clave primaria de la relación consiste en la unión de la clave primaria del conjunto de entidades fuertes y el discriminante del conjunto de entidades débil.

Conjunto de relaciones: La unión de las claves primarias de los conjuntos de entidades relacionados se transforma en una superclave de la relación. Si la relación es de varios a varios, esta superclave es también la clave primaria.

6) Tablas combinadas: un conjunto binario de relaciones de varios a uno entre *A* y *B* puede representarse mediante una tabla que consista en los atributos de *A* y en los atributos (si hay alguno) del conjunto de relaciones. La clave primaria de la entidad «varios» se transforma en la clave primaria de la relación (es decir, si el conjunto de relaciones es de varios a uno entre *A* y *B*, la clave primaria de A es la clave primaria de la relación). Para los conjuntos de relaciones de uno a uno la relación se construye igual que en el conjunto de relaciones de varios a uno. Sin embargo, cualquiera de las claves primarias del conjunto de entidades puede elegirse como clave primaria de la relación, dado que ambas son claves candidatas

### 2.3.3.2. Lenguajes de consulta.

Un lenguaje de consulta es un lenguaje en el que un usuario solicita información de la base de datos. Estos lenguajes suelen ser de un nivel superior que el de los lenguajes de programación habituales. Los lenguajes de consulta pueden clasificarse como procedimentales o no procedimentales. En los lenguajes procedimentales el usuario instruye al sistema para que lleve a cabo una serie de operaciones en la base de datos para calcular el resultado deseado. En los

lenguajes no procedimentales el usuario describe la información deseada sin dar un procedimiento concreto para obtener esa información.
La mayor parte de los sistemas comerciales de bases de datos relacionales ofrecen un lenguaje de consulta que incluye elementos de los enfoques procedimental y no procedimental.

### 2.3.4. Algebra Relacional.

El álgebra relacional es un lenguaje de consulta procedimental. Consta de un conjunto de operaciones que toman como entrada una o dos relaciones y producen como resultado una nueva relación. Las operaciones fundamentales del álgebra relacional son selección, proyección, unión, diferencia de conjuntos, producto cartesiano y renombramiento.

Las operaciones selección, proyección y renombramiento se denominan operaciones unarias porque operan sobre una sola relación. Las otras tres operaciones operan sobre pares de relaciones y se denominan, por lo tanto, operaciones binarias.

#### 2.3.4.1. La Operación selección.

La operación selección selecciona tuplas que satisfacen un predicado dado. Se utiliza la letra griega sigma minúscula (σ) para denotar la selección. El predicado aparece como subíndice de σ. La relación del argumento se da entre paréntesis a continuación de σ.
Sintaxis:
$\sigma_{condición}(R) = \{ t \in R : condición(t) \text{ es cierto} \}$

En SQL
$\sigma_{condición}(R)$

*SELECT * FROM R WHERE condición*

#### 2.3.4.2. La operación proyección.

La operación proyección es una operación unaria que devuelve su relación de argumentos, excluyendo algunos argumentos. Dado que las relaciones son conjuntos, se eliminan todas las filas duplicadas. La proyección se denota por la letra griega mayúscula pi (π). Se crea una lista de los atributos que se desea que aparezcan en el resultado como subíndice de π. La relación de argumentos se escribe a continuación entre paréntesis.

Sintaxis:
$\pi_{A1,A2,...,An}(R) = \{ t[A_1,A_2,...,A_n] : t \in R \}$

En SQL
$\pi_{A1,A2,...,An}(R)$

SELECT $A_1$,$A_2$,...,$A_n$ FROM R

### 2.3.4.3. La operación unión.

La unión de dos relaciones *R y S*, es otra relación que contiene las tuplas que están en *R* o en *S, o* en ambas eliminándose las tuplas duplicadas.

*R y S* deben ser unión- compatible, es decir, definidas sobre el mismo conjunto de atributos.

$R \cup S = \{t/t \in R \text{ ó } t \in S\}$

### 2.3.4.4. La operación diferencia de conjuntos

La operación diferencia de conjuntos, denotada por –, permite buscar las tuplas que estén en una relación pero no en la otra. La expresión *r – s* da como resultado una relación que contiene las tuplas que están en *r* pero no en *s*.

*Ejemplo: sean r y s dos relaciones denotadas por:*

*r = {P5, P7, P9}*
*s = {P7, P3}*

*r – s = {P5, P7, P9} – {P7, P3}*
*{P5, P9}*

### 2.3.4.5. La operación producto Cartesiano.

La operación producto cartesiano, denotada por un aspa (·), permite combinar información de cualesquiera dos relaciones. El producto cartesiano de las relaciones $r_1$ y $r_2$ como $r_1 \cdot r_2$. Recuérdese que las relaciones se definen como subconjuntos del producto cartesiano de un conjunto de dominios. A partir de esta definición ya se debe tener una intuición sobre la definición de la operación producto cartesiano. Sin embargo, dado que el mismo nombre de atributo puede aparecer tanto en $r_1$ como en $r_2$,
Hay que crear un esquema de denominaciones para distinguir entre ambos atributos. En este caso se logra adjuntando al atributo el nombre de la relación de la que proviene originalmente.

Ejemplo: sean $r_1$ y $r_2$ dos relaciones denotadas por:

$r_1 = \{s, t\}$
$r_2 = \{u, v, w\}$

$r_1 \cdot r_2 = \{s,t\} \cdot \{u,v,w\}$

{(s, u), (s, v), (s, w), (t, u), (t, v), (t, w) }

*En SQL*

$r_1 \cdot r_2$
SELECT * FROM $r_1$, $r_2$

### 2.3.4.6. La operación renombramiento.

A diferencia de las relaciones de la base de datos, los resultados de las expresiones de álgebra relacional no tienen un nombre que se pueda utilizar para referirse a ellas. Resulta útil poder ponerles nombre; el operador renombramiento, denotado por (ρ), permite realizar esta tarea. Dada una expresión *E* del álgebra relacional, la expresión

$$\rho_x (E)$$

Devuelve el resultado de la expresión *E* con el nombre *x*. Las relaciones *r* por sí mismas se consideran expresiones (triviales) del álgebra relacional. Por tanto, también se puede aplicar la operación renombramiento a una relación *r* para obtener la misma relación con un nombre nuevo.
Otra forma de la operación renombramiento es la siguiente. Supóngase que una expresión del álgebra relacional *E* tiene aridad *n*. Por tanto, la expresión

$$\rho_x (A_1, A_2, \ldots A_n) (E)$$

Devuelve el resultado de la expresión *E* con el nombre *x* y con los atributos con el nombre cambiado a *A*1, *A*2, …, *An.*

### 2.3.5. Operaciones del álgebra relacional extendida.

Las operaciones básicas del álgebra relacional se han ampliado de varias maneras. Una ampliación sencilla es permitir operaciones aritméticas como parte de la proyección. Una ampliación importante es permitir operaciones de agregación, como el cálculo de la suma de los elementos de un conjunto, o su media. Otra ampliación importante es la operación reunión externa, que permite a las expresiones del álgebra relacional trabajar con los valores nulos que modelan la información que falta.

### 2.3.5.1. Proyección generalizada.

La operación proyección generalizada amplía la operación proyección permitiendo que se utilicen funciones aritméticas en la lista de proyección. La operación proyección generalizada tiene la forma

$$\pi F_1, F_2, \ldots, F_n (E)$$

donde *E* es cualquier expresión del álgebra relacional y $F_1$, $F_2$, …, $F_n$ son expresiones aritméticas que incluyen constantes y atributos en el esquema de *E*. Como caso especial la expresión aritmética puede ser simplemente un atributo o una constante.

### 2.3.5.2. Funciones de agregación.

Las funciones de agregación son funciones que toman una colección de valores y devuelven como resultado un único valor

La operación reunión externa es una ampliación de la operación reunión para trabajar con la información que falta

La reunión externa por la izquierda (⟕) toma todas las tuplas de la relación de la izquierda que no coincidan con ninguna tupla de la relación de la derecha, las rellena con valores nulos en todos los demás atributos de la relación de la derecha y las añade al resultado de la reunión natural

La reunión externa por la derecha (⟖) es simétrica de la reunión externa por la izquierda. Las tuplas de la relación de la derecha que no coincidan con ninguna tupla de la relación de la izquierda se rellenan con valores nulos y se añaden al resultado de la reunión natural.

La reunión externa completa (⟗ ) realiza estas dos operaciones, rellenando las tuplas de la relación de la izquierda que no coincidan con ninguna tupla de la relación de la derecha y las tuplas de la relación de la derecha que no coincidan con ninguna tupla de la relación de la izquierda, y añadiéndolas al resultado de la reunión.

### 2.3.5.3. Valores nulos.

En este apartado se define la forma en que las diferentes operaciones del álgebra relacional tratan los valores nulos y las complicaciones que surgen cuando los valores nulos participan en las operaciones aritméticas o en las comparaciones. Como se verá, a menudo hay varias formas de tratar los valores nulos y, como resultado, las siguientes definiciones pueden ser a veces arbitrarias.

Las operaciones y las comparaciones con valores nulos se deberían evitar siempre que sea posible.
Dado que el valor especial nulo indica «valor desconocido o no existente», cualquier operación aritmética (como +, –, * y /) que incluya valores nulos debe devolver un valor nulo.
De manera similar, cualquier comparación (como <, <=, >, >= y ≠) que incluya un valor nulo se evalúa al valor especial desconocido; no se puede decir si el resultado de la comparación es cierto o falso, así que se dice que el resultado es el nuevo valor lógico desconocido.

Las comparaciones que incluyan nulos pueden aparecer dentro de expresiones booleanas que incluyan las operaciones y (conjunción), o (disyunción) y no (negación).
Se debe definir la forma en que estas operaciones tratan el valor lógico desconocido.

**y**: (cierto **y** desconocido) = desconocido; (falso **y** desconocido) = falso; (desconocido **y** desconocido) = desconocido.
**o**: (cierto **o** desconocido) = cierto; (falso **o** desconocido) = desconocido; (desconocido **o** desconocido) = desconocido.
**no**: (**no** desconocido) = desconocido.

Ahora es posible describir la forma en que las diferentes operaciones del álgebra relacional tratan los valores nulos. Nuestras definiciones siguen las usadas en el lenguaje SQL.

- Select: la operación selección evalúa el predicado P en σP(E) sobre cada tupla de E. Si el predicado devuelve el valor cierto, se añade t al resultado. En caso contrario, si el predicado devuelve desconocido o falso, t no se añade al resultado.
- Reunión: las reuniones se pueden expresar como un producto cartesiano seguido de una selección. Por tanto, la definición de la forma en que la selección trata los nulos también define la forma en que la operación reunión trata los nulos. En una reunión natural r ⋈ s se puede observar de la definición anterior que si dos tuplas, $t_r$ Є r y $t_s$ Є s, tienen un valor nulo en un atributo común, entonces las tuplas no casan.
- Proyección: la operación proyección trata los nulos como cualquier otro valor al eliminar duplicados. Así, si dos tuplas del resultado de la proyección son exactamente iguales, y ambos tienen nulos en los mismos campos, se tratan como duplicados. La decisión es un tanto arbitraria porque sin saber cuál es el valor real no se sabe si los dos valores nulos son duplicados o no.
- Unión, intersección, diferencia: estas operaciones tratan los valores nulos al igual que la operación proyección; tratan las tuplas que tienen los mismos valores en todos los campos como duplicados incluso si algunos de los campos tienen valores nulos en ambas tuplas. El comportamiento es un tanto arbitrario, especialmente en el caso de la intersección y la diferencia, dado que no se sabe si los valores reales (si existen) representados por los nulos son los mismos.
- Reunión externa: las operaciones de reunión externa se comportan como las operaciones reunión, excepto sobre las tuplas que no aparecen en el resultado. Estas tuplas se pueden añadir al resultado añadiendo nulos.

### 2.3.6. Normalización.

La normalización [MySQL hispano, 03] es el proceso mediante el cual se transforman datos complejos a un conjunto de estructuras de datos más pequeñas, que además de ser más simples y más estables, son más fáciles de mantener. También se puede entender la normalización como una serie de reglas que sirven para ayudar a los diseñadores de bases de datos a desarrollar un esquema que minimice los problemas de lógica. Cada regla está basada en la que le antecede. La normalización se adoptó porque el viejo estilo de poner todos los datos en un solo lugar, como un archivo o una tabla de la base de datos, era ineficiente y conducía a errores de lógica cuando se trataban de manipular los datos.

La normalización también hace las cosas fáciles de entender. Los seres humanos tenemos la tendencia de simplificar las cosas al máximo. Lo hacemos con casi todo, desde los animales hasta con los automóviles. Vemos una imagen de gran tamaño y la hacemos más simple agrupando cosas similares juntas. Las guías que la normalización provee crean el marco de referencia para simplificar una estructura de datos compleja.

Otra ventaja de la normalización de base de datos es el consumo de espacio. Una base de datos normalizada ocupa 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.

El proceso de normalización tiene un nombre y una serie de reglas para cada fase. Esto puede parecer un poco confuso al principio, pero poco a poco se va entendiendo el proceso, así como las razones para hacerlo de esta manera.

Hay algunas reglas en la normalización de una base de datos. Cada regla se denomina una "forma normal". Si se cumple la primera regla, se dice que la base de datos está en la "primera forma normal". Si se cumplen las tres primeras reglas, la base de datos se considera que está en la "tercera forma normal". Aunque son posibles otros niveles de normalización, la tercera forma normal se considera el máximo nivel necesario para la mayor parte de las aplicaciones.

Al igual que con otras muchas reglas y especificaciones formales, en los escenarios reales no siempre se cumplen los estándares de forma perfecta. En general, la normalización requiere tablas adicionales y algunos clientes consideran éste un trabajo considerable. Si decide infringir una de las tres primeras reglas de la normalización, asegúrese de que su aplicación se anticipa a los problemas que puedan aparecer, como la existencia de datos redundantes y de dependencias incoherentes.

### 2.3.6.1. Grados de normalización.

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 (ver Figura 2.7).

Cuando una base de datos se conforma a un nivel, se considera normalizada a esa forma de normalización. No siempre es una buena idea tener una base de datos conformada en el nivel más alto de normalización, puede llevar a un nivel de complejidad que pudiera ser evitado si estuviera en un nivel más bajo de normalización. En la tabla siguiente se describe brevemente en qué consiste cada una de las reglas, y posteriormente se explican a detalle.

| Regla | Descripción |
|---|---|
| Primera Forma Normal (1FN) | Incluye la eliminación de todos los grupos repetidos. |
| Segunda Forma Normal (2FN) | Asegura que todas las columnas que no son llave sean completamente dependientes de la llave primaria (PK). |
| Tercera Forma Normal (3FN) | Elimina cualquier dependencia transitiva. Una dependencia transitiva es aquella en la cual las columnas que no son llave son dependientes de otras columnas que tampoco son llave |

Figura 2.7: Niveles de Normalización Básicos

#### 2.3.6.1.1. Primera Forma Normal.

La regla de la Primera Forma Normal establece que las columnas repetidas deben eliminarse y colocarse en tablas separadas. Poner la base de datos en la Primera Forma Normal resuelve el problema de los encabezados de columna múltiples. Muy a menudo, los diseñadores de bases de datos inexpertos harán algo similar a la tabla no normalizada. Una y otra vez, crearán columnas que representen los mismos datos. La normalización ayuda a clarificar la base de datos y a organizarla en partes más pequeñas y más fáciles de entender. En lugar de tener que entender una tabla gigantesca y monolítica que tiene muchos diferentes aspectos, sólo tenemos que entender los objetos pequeños y más tangibles, así como las relaciones que guardan con otros objetos también pequeños.

### 2.3.6.1.2. Segunda Forma Normal.

La regla de 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 llave primaria de la tabla para identificarlos. Una vez alcanzado el nivel de la Segunda Forma Normal, se controlan la mayoría de los problemas de lógica. Podemos insertar un registro sin un exceso de datos en la mayoría de las tablas.

### 2.3.6.1.3. Tercera Forma Normal.

Una tabla está normalizada en esta forma si todas las columnas que no son llave son funcionalmente dependientes por completo de la llave primaria y no hay dependencias transitivas. Comentamos anteriormente que una dependencia transitiva es aquella en la cual existen columnas que no son llave que dependen de otras columnas que tampoco son llave. Cuando las tablas están en la Tercera Forma Normal se previenen errores de lógica cuando se insertan o borran registros. Cada columna en una tabla está identificada de manera única por la llave primaria, y no debe haber datos repetidos. Esto provee un esquema limpio y elegante, que es fácil de trabajar y expandir. Un dato sin normalizar no cumple con ninguna regla de normalización.

### 2.3.6.1.5. Otras formas de normalización.

La cuarta forma normal, también llamada Forma normal de Boyce Codd (BCNF, Boyce Codd Normal Form), y la quinta forma normal existen, pero rara vez se consideran en un diseño real. Si no se aplican estas reglas, el diseño de la base de datos puede ser menos perfecto, pero no debería afectar a la funcionalidad.

### 2.3.7. SQL.

SQL (*Structured Query Language*, Lenguaje estructurado de consultas) usa una combinación de álgebra relacional y construcciones del cálculo relacional. Aunque el lenguaje SQL se considere un lenguaje de consultas, contiene muchas otras capacidades además de la consulta en bases de datos. Incluye características para definir la estructura de los datos, para la modificación de los datos en la base de datos y para la especificación de restricciones de seguridad.
A continuación se presentan las construcciones y conceptos fundamentales de SQL.

### 2.3.71. Estructura Básica.

Una base de datos relacional consiste en un conjunto de relaciones, a cada una de las cuales se le asigna un nombre único.
SQL permite el uso de valores nulos para indicar que el valor o bien es desconocido, o no existe. Se fijan criterios que permiten al usuario especificar a qué atributos no se puede asignar valor nulo.

La estructura básica de una expresión SQL consiste en tres cláusulas: *select, from y where.*

- La cláusula *select* corresponde a la operación proyección del álgebra relacional. Se usa para listar los atributos deseados del resultado de una consulta.
- La cláusula *from* corresponde a la operación producto cartesiano del álgebra relacional. Lista las relaciones que deben ser analizadas en la evaluación de la expresión.
- La cláusula *where* corresponde al predicado selección del álgebra relacional. Es un predicado que engloba a los atributos de las relaciones que aparecen en la cláusula *from.*

Un hecho histórico desafortunado es que el término *select* tiene un significado diferente en SQL que en el álgebra relacional. A continuación se resaltan las diferentes interpretaciones, a fin de minimizar la posible confusión. Una consulta típica en SQL tiene la forma

select $A_1, A_2, \ldots, A_n$
from $r_1, r_2, \ldots, r_m$
where $P$

Cada $A_i$ representa un atributo, y cada $r_i$ una relación. $P$ es un predicado. La consulta es equivalente a la expresión del álgebra relacional
$\pi\, A_1, A_2, \ldots, A_n\, (\sigma P\, (r_1 \times r_2 \times \ldots \times r_m\, ))$
Si se omite la cláusula *where,* el predicado P es cierto. Sin embargo, con diferencia a la expresión del álgebra relacional, el resultado de la consulta SQL puede contener varias copias de algunas tuplas.

SQL forma el producto cartesiano de las relaciones incluidas en la cláusula *from,* lleva a cabo la selección del álgebra relacional usando el predicado de la cláusula *where* y entonces proyecta el resultado sobre los atributos de la cláusula *select.* En la práctica, SQL puede convertir la expresión en una forma equivalente que puede ser procesada más eficientemente

**2.3.7.2. ERWIN .**

ERwin Data Modeler (CA ERwin DM) es el producto insignia de la familia, que ayuda a las organizaciones a crear modelos de datos que respalden tanto las necesidades de negocios de alto nivel como el diseño y el uso de la base de datos a nivel de la implementación a través de una amplia gama de plataformas. Las tecnologías de soporte incluyen: creación de perfiles de datos para comparar datos de instancias en vivo con proyectos de modelos, modelado de procesos para alinear los procesos de negocio con los activos de datos, validación de modelos,

generación de reportes, administración de metadatos, y control de versiones y colaboració.

ERwin *Data Model Validator* (CA ERwin DMV) automatiza la tarea de comparación del esquema de sus bases de datos con relación a las reglas relacionales. Debido a que cada cambio en el esquema puede afectar negativamente o hasta corromper el proyecto de base de datos, CA ERwin DMV le permite validar el esquema antes que los cambios sean implementadas, minimizando los efectos adversos de las alteraciones no autorizadas. Los recursos de validación de modelos incluyen:

- Diagnósticos y elaboración de reportes integrales: Analizan la estructura de los datos del esquema, claves, índices, columnas de campos y relaciones, buscando violaciones de la teoría relacional, y generan una representación gráfica de la estructura completa de la base de datos, incluyendo referencias cruzadas de columnas y listas de relaciones. Además, los diagnósticos pueden ser personalizados para mostrar solamente la categoría o nivel de gravedad de aquellas áreas más importantes para su organización.
- Aislamiento de discrepancias en el proyecto: Proporciona reportes detallados que ayudan a aumentar la productividad a través de la aceleración del proceso de revisión del proyecto, utilizando un innovador recurso "*show m*e", que aísla problemas específicos del proyecto en modelos complejos de bases de datos, eliminando la tarea de identificar o validar manualmente problemas en una base de datos o un subconjunto de modelos.

Recurso "*Teach me*": Explica la teoría por detrás de las violaciones de reglas relacionales y revela el impacto de las opciones o modificaciones del proyecto antes que ocurran. Al enseñar a los modeladores de datos sobre los efectos de sus decisiones de proyecto, CA ERwin DMV permite la construcción de bases de datos de alta calidad y simplifica el entrenamiento del personal nuevo.

AllFusion ERwin Data Modeler es una herramienta de diseño de base de datos que ayuda a los usuarios a diseñar, generar y mantener alta calidad de las aplicaciones de base de datos de altas prestaciones. AllFusion ERwin Data Modeler permite al usuario visualizar la estructura correcta, elementos claves y el diseño optimizado de su base de datos, desde los requerimientos de un modelo lógico de información y reglas de negocio que definen la base de datos, a un modelo físico optimizado para las características específicas de la base de datos seleccionada.
AllFusion ERwin Data Modeler automáticamente genera tablas y miles de líneas de procedimientos almacenados y códigos disparadores para las base de datos líderes.

Su tecnología de "comparación completa" permite el desarrollo interactivo, de forma tal que los modelos están siempre sincronizados con la base de datos del usuario. Al integrarse con entornos de desarrollo líderes, AllFusion ERwin Data Modeler también acelera la creación de aplicaciones centralizadas en datos.

### 2.3.8. SQLSERVER 2008.

Microsoft SQL Server 2008 es un sistema de Gestión de Bases de datos relacionales (SGDBR o RDBMS*: Relational Management System*) Cliente/Servidor de alto rendimiento. Se ha diseñado para admitir un elevado volumen de procesamiento de transacciones (como la entrada de pedidos en línea, inventarios, contabilidad o facturación). Además de aplicaciones de almacén de datos y de ayuda en la toma de decisiones (como la aplicación de análisis de ventas).

#### 2.3.8.1. Bases de Datos.

Dicho en términos sencillos, una base de datos de Microsoft SQL Server es una colección de objetos que contiene y maneja datos.

##### 2.3.8.1.1. Creación de Bases de Datos.

La forma más sencilla de crear una base de datos consiste en utilizar el administrador corporativo de SQL Server, que proporciona un entorno gráfico para los comandos y los procedimientos almacenados de Transct-SQL que crean realmente la base de datos y establecen sus propiedades. En la creación de una base de datos con el administrador corporativo antes mencionado representa el comando de CREATE DATABASE de Transact-SQL para crear una nueva base de datos de usuario. Solo alguien con la función sysadmin o un usuario le haya concedido permiso CREATE DATEBASE podrá emitir el comando CREATE DATABASE.

Una nueva Base de Datos de usuario debe tener un tamaño de 1 Mb o superior, y el tamaño del archivo de datos primeros debe ser al menos del tamaño del archivo de datos primario de la base de datos. Prácticamente todos los argumentos posibles del comando CREATE DATABASE tienen valores predeterminados, de forma que es posible crear una base de datos utilizando un formato sencillo de CREATE DATABASE.

#### 2.3.8.2. Tablas.

En términos sencillos una tabla es una colección de datos sobre una entidad (persona, lugar o cosa) especifica, que contiene un número discreto de atributos designados (por ejemplo, cantidad o tipo). Las tablas están en el corazón de SQL Server y del modelo relacional en general. Las tablas son fáciles de entender, ya que son prácticamente que se utiliza de manera cotidiana. En SQL Server, una tabla suele denominarse tabla de base, para hacer énfasis sobre dónde se almacenan los datos. La utilización de tabla de base también distingue la tabla de una vista (*view*), una tabla virtual que es una consulta interna que hace referencia a una o varias tablas base.

Los atributos de los datos de una tabla (como color, tamaño, cantidad, fecha de pedido y nombre del proveedor) toman la forma de columnas con nombre en la tabla. Cada instancia de los datos en una tabla viene representada por una única entrada, o fila (llamada formalmente tupla).en una base de datos verdaderamente relacional, cada fila de una tabla es exclusiva, y tiene un identificador exclusivo denominado clave primaria (SQL Server, se acuerdo con el estándar ANSI SQL, no exige que una fila sea exclusiva o que se declare una clave primaria. Sin embargo como ambos conceptos son fundamentales en el modelo relacional deberán implementarse siempre).

La mayor parte de las tablas suelen estar relacionadas con otras tablas. Por ejemplo en un sistema entrada de pedidos, la tabla pedidos probablemente tenga una columna número_cliente donde se guarda el número del cliente que ha realizado un pedido; número_cliente también aparece en la tabla cliente. Asumiendo que número_cliente es un identificador exclusivo o la clave primaria, de la tabla cliente, se establece una relación de clave externa (foreign key) mediante la cual pueden combinarse y posteriormente la tabla pedidos y cliente.

### 2.3.8.2.1. Creación de Tablas.

Para crear una Tabla, SQL Server utiliza la sintaxis CREATE TABLE estándar del ANSI SQL, el administrador corporativo de SQL Server ofrece un editor de tablas del tipo <<rellenar los huecos>> que hará la vida más fácil.
A nivel básico para crear una tabla hace falta un poco más que saber el nombre que se le quiere dar que columnas va a contener y que rango de valores(dominio) deberá almacenar cada columna, esta es la sintaxis básica para crear la tabla cliente, con tres columnas de tipo carácter(char) de longitud fija.

```
CREATE TABLE cliente
{
Nombre char (30);
Télefono char (12);
Id_emp char (4);
}
```

### 2.3.8.2.2. Asignar Nombres a Tablas y Columnas.

Una tabla siempre se crea dentro de una base de datos y es propiedad de un usuario concreto. Habitualmente el propietario es el usuario que ha creado la tabla. Una base de datos puede contener varias tablas con el mismo nombre, siempre que las tablas tengan distintos propietarios. El nombre de la tabla tiene tres partes, en el siguiente formato:

Basededatos.propietario.nombredetabla.

Por ejemplo supongamos que un usuario (con el nombre de usuario Kelen ) ha creado una tabla cliente de ejemplo en la base de datos de ejemplo *bubs*. La tabla de este usuario tendrá el nombre de tres partes *bubs.kelen.cliente* (se esté usuario es también el propietario de la base de datos, el nombre de su tabla será *pubs.dbo.cliente*, porque *dbo* es el nombre de usuario especial para el propietario de la base de datos en todas bases de datos).

Los nombres de las columnas deben ser descriptivos y, como los va a utilizar con frecuencia, es mejor evitar extenderse innecesariamente. El nombre de la columna(o de cualquier objeto SQL Server como una tabla o una vista) puede ser el que se refiera, siempre que se ajuste a las reglas de SQL Server para los identificadores: debe constar de 1 a 128 letras, números o símbolos.

### 2.3.8.3. Tipos de Datos.

Un tipo de datos [ TiposDatos SQLServer. ] es un atributo que especifica el tipo de datos que el objeto puede contener: datos de enteros, datos de caracteres, datos de moneda, datos de fecha y hora, cadenas binarias, etc.

SQL Server proporciona un gran número de tipos de datos, algunos de los cuales se explicarán en esta sección. Seleccionar el tipo de datos adecuado es simplemente una cuestión de asociar el dominio de valores que necesita almacenar con el tipo de datos correspondiente. Al elegir tipos de datos deberá evitar desperdiciar espacios de almacenamiento, a la vez proporciona espacio suficiente para el rango de valores posibles durante la vida de la aplicación.

Los tipos de datos de SQL Server se organizan en las siguientes categorías:

- Numéricos exactos ( bigint,numeric bit, smallint, decimal, smallmoney, int, tinyint, money).
- Numéricos aproximados ( float, real).
- Cadenas de caracteres ( char, varchar, text).
- Cadenas de caracteres Unicode ( nchar, nvarchar, ntext).
- Cadenas binarias ( binary, varbinary, image).
- fecha y hora ( date, datetimeoffset, datetime2, smalldatetime, datetime, time).
- Otros tipos de datos (cursor, timestamp, hierarchyid, uniqueidentifier, sql_variant, xml, table).

En SQL Server, según las características de almacenamiento, algunos tipos de datos están designados como pertenecientes a los siguientes grupos:

- Tipos de datos de valores grandes: varchar(max), nvarchar(max) y varbinary(max)
- Tipos de datos de objetos grandes: text, ntext, image, varchar(max), nvarchar(max), varbinary(max) y xml

SQL Server 2008, define nuevos tipos de datos que ya son (o serán) novedades tenemos a los siguientes: HIERARCHY ID (para almacenar valores de nodos en un jerarquía), FILESTREAM (Almacena grandes ficheros de datos binarios no estructurados directamente en un sistema de ficheros NTFS: documentos, imágenes, etc,), tipos para almacenar datos espaciales como GEOGRAPHY, GEOMETRY, así como los tipos de datos TIME (3 a 5 bytes de tamaño) y DATE (3bytes), DATETIME2 (6 a 8 bytes), DATETIMEOFFSET (8 a 10 bytes), entre otros más..

### 2.3.8.4. Consultar Datos.

No tendría caso alguno guardar datos si no se quieran recuperar, las bases de datos relacionales consiguieron una amplia aceptación principalmente porque les permitían a los usuarios consultar o acceder a los datos de manera sencilla, sin utilizar trayectos de recorrido predefinidos y rígidos. SQL (*Structured Query Language* - Lenguaje de consulta estructurado).

#### 2.3.8.4.1. SELECT.

La instrucción SELECT (seleccionar) es el comando SQL que más se utiliza, y es la forma fundamental de consultar datos, Recupera filas de la base de datos y habilita la selección de una o varias filas o columnas de una o varias tablas. La sintaxis completa de la instrucción SELECT es compleja, aunque las cláusulas principales se pueden resumir del modo siguiente:

[ WITH <common_table_expression>]
SELECT *select_list* [ INTO *new_table* ]
[ FROM *table_source* ] [ WHERE *search_condition* ]
[ GROUP BY *group_by_expression* ]
[ HAVING *search_condition* ]
[ ORDER BY *order_expression* [ ASC | DESC ] ]

Los operadores UNION, EXCEPT e INTERSECT se pueden utilizar entre consultas para combinar o comparar resultados en un conjunto de resultados. A continuación un ejemplo.

En este ejemplo solo se devuelven las filas de *Product* que tienen una línea de productos de R y cuyo valor correspondiente a los días para fabricar es inferior a 4.

```
USE AdventureWorks2008R2;
GO
SELECT Name, ProductNumber, ListPrice AS Price
FROM Production.Product
WHERE ProductLine = 'R'
AND DaysToManufacture < 4
ORDER BY Name ASC;
GO
```

### 2.3.8.5. Modificar Datos.

En la sección anterior vimos como consultar datos, en esta sección veremos la inserción, actualización y eliminación de datos en tablas. Veremos los comandos básicos para estas operaciones pero no nos concentraremos en los comportamientos específicos de SQL Server.

A continuación describimos las Operaciones básicas de modificar datos (INSERT, UPDATE, DELETE).

#### 2.3.8.5.1. INSERT.

Utilice la instrucción INSERT para especificar valores directamente o desde una subconsulta. La instrucción INSERT agrega una o más filas nuevas a una tabla. Tratada de forma simplificada, INSERT tiene el siguiente formato:

INSERT [INTO] table_or_view [(column_list)] data_values

La instrucción INSERT inserta data_values como una o más filas en la tabla o vista especificada. column_list es una lista separada por comas de los nombres de columnas que se pueden utilizar para especificar las columnas para las que se suministran datos. Si no se especifica column_list, todas las columnas de la tabla o vista reciben datos.
Cuando column_lis*t* no especifica todas las columnas de la tabla o vista, se inserta el valor predeterminado, si se ha definido alguno para la columna, o un valor de NULL en aquellas columnas que no se hayan especificado en la lista. Todas las columnas no especificadas en la lista de columnas deben permitir valores NULL o tener asignado un valor predeterminado.

#### 2.3.8.5.2. UPDATE.

La instrucción UPDATE puede cambiar los valores de filas individuales, grupos de filas o todas las filas de una tabla o vista. También se puede utilizar para actualizar las filas de un servidor remoto utilizando un nombre de servidor vinculado o las funciones OPENROWSET, OPENDATASOURCE y OPENQUERY, siempre que el proveedor OLE DB utilizado para obtener acceso al servidor remoto admita actualizaciones. Una instrucción UPDATE que haga referencia a una tabla o vista sólo puede cambiar los datos de una tabla a la vez.
La instrucción UPDATE tiene las siguientes cláusulas principales:

- SET: Contiene una lista separada por comas de las columnas que deben actualizarse y el nuevo valor de cada columna con el formato column_name = expression. El valor suministrado por las expresiones incluye elementos tales como constantes, valores seleccionados de una columna de otra tabla o vista, o valores calculados por una expresión compleja. Para obtener más información, vea Cambiar datos con la cláusula SET.
- FROM: Identifica las tablas o vistas que suministran los valores de las expresiones de la cláusula SET, y las condiciones de combinación opcional

entre las tablas o vistas de origen. Para obtener más información, vea Cambiar datos con la cláusula FROM.

- WHERE: Especifica la condición de búsqueda que define las filas de las tablas y vistas de origen que están calificadas para proporcionar valores para las expresiones de la cláusula SET. Para obtener más información, vea Cambiar datos con la cláusula WHERE.

En el siguiente ejemplo se utiliza la instrucción UPDATE para aumentar un 10 por ciento el precio de todos los productos de *AdventureWorks* asociados al modelo de producto 37:

```
USE AdventureWorks;
GO
UPDATE AdventureWorks.Production.Product
SET ListPrice = ListPrice * 1.1
WHERE ProductModelID = 37;
GO
```

**2.3.8.5.3. DELETE.**

La instrucción DELETE quita una o varias filas de una tabla o vista. A continuación se expone una forma simplificada de la sintaxis de DELETE:

```
DELETE table_or_view
FROM table_sources
WHERE search_condition
```

El parámetro table_or_view indica la tabla o vista de la que se van a eliminar las filas. Se eliminarán todas las filas de table_or_view que reúnan los requisitos de la condición de búsqueda de la cláusula WHERE. Si no se especifica ninguna cláusula WHERE, se eliminarán todas las filas de table_or_view. La cláusula FROM especifica tablas o vistas y condiciones de combinación adicionales que los predicados de la condición de búsqueda de la cláusula WHERE pueden utilizar para calificar las filas que se eliminarán de table_or_view. Las filas no se eliminan de las tablas mencionadas en la cláusula FROM, sólo de la tabla mencionada en table_or_view.

Las tablas de las que se quitan todas las filas permanecen en la base de datos. La instrucción DELETE sólo elimina filas de la tabla, pero la tabla en sí se debe quitar de la base de datos mediante la instrucción DROP TABLE.

En el ejemplo siguiente se eliminan todas las filas de la tabla ProductCostHistory en las que el valor de la columna StandardCost es superior a 1000.00.

```
USE AdventureWorks2008R2;
GO
DELETE FROM Production.ProductCostHistory
```

```
WHERE StandardCost > 1000.00;
GO
```

## 2.4. WEB.

La web, tal y como la conocemos hoy día, ha permitido un flujo de comunicación global a una escala sin precedentes en la historia humana. La propagación de información a través de la web (vía Internet) no está limitada y puede ser buscada más fácil y eficientemente gracias a su carácter virtual.

La web es el medio de mayor difusión de intercambio personal aparecido en la Historia de la Humanidad. Esta plataforma ha permitido a los usuarios interactuar con muchos más grupos de personas dispersas en tiempo y espacio.

Entre las tecnologías que desarrollos que posibilitan la web:

CSS Para tener una mejor organización de la interfaz del usuario, JavaScript y DOM hacen posible una interfaz amigable, dinámica para el usuario así como también para darle una mejor funcionalidad al sistema.

### 2.4.1. CSS.

CSS [Eguíluz, 09] es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. CSS (*Cascading Style Sheets* - hojas de estilo en cascada) es la mejor forma de separar los contenidos y su presentación y es imprescindible para crear páginas web complejas.

Separar la definición de los contenidos y la definición de su aspecto presenta numerosas ventajas, ya que obliga a crear documentos HTML/XHTML bien definidos y con significado completo (también llamados "documentos semánticos").

Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes.

Al crear una página web, se utiliza en primer lugar el lenguaje HTML/XHTML para marcar los contenidos, es decir, para designar la función de cada elemento dentro de la página: párrafo, titular, texto destacado, tabla, lista de elementos, etc.
Una vez creados los contenidos, se utiliza el lenguaje CSS para definir el aspecto de cada elemento: color, tamaño y tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página, etc.

Dentro de las ventajas principales de usar Hojas de estilo son:

- El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño
- Se tiene una mejor organización de los elementos de un documento HTML, como divisiones, márgenes, párrafos, etc.
- Control centralizado de la presentación de un sitio web completo, con lo que se agiliza y facilita de forma considerable la actualización del mismo.

### 2.4.2 DOM.

DOM [Eguíluz, 09], es una de las innovaciones que más ha influido en el desarrollo de las páginas web dinámicas y de las aplicaciones web más complejas. Permite a los programadores web acceder y manipular las páginas XHTML como si fueran documentos XML. De hecho, DOM (*Document Object Model* - Modelo de Objetos del Documento) se diseñó originalmente para manipular de forma sencilla los documentos XML.

A pesar de sus orígenes, DOM se ha convertido en una utilidad disponible para la mayoría de lenguajes de programación (Java, PHP, JavaScript) y cuyas únicas diferencias se encuentran en la forma de implementarlo.

#### 2.4.2.1. Árbol de nodos.

Una de las tareas habituales en la programación de aplicaciones web con JavaScript consiste en la manipulación de las páginas web. De esta forma, es habitual obtener el valor almacenado por algunos elementos (por ejemplo los elementos de un formulario), crear un elemento (párrafos, , etc.) de forma dinámica y añadirlo a la página, aplicar una animación a un elemento (que aparezca/desaparezca, que se desplace, etc.).

Todas estas tareas habituales son muy sencillas de realizar gracias a DOM. Sin embargo, para poder utilizar las utilidades de DOM, es necesario "transformar" la página original. Una página HTML normal no es más que una sucesión de caracteres, por lo que es un formato muy difícil de manipular. Por ello, los navegadores web transforman automáticamente todas las páginas web en una estructura más eficiente de manipular.

Esta transformación la realizan todos los navegadores de forma automática y nos permite utilizar las herramientas de DOM de forma muy sencilla. El motivo por el que se muestra el funcionamiento de esta transformación interna es que condiciona el comportamiento de DOM y por tanto, la forma en la que se manipulan las páginas.

DOM transforma todos los documentos XHTML en un conjunto de elementos llamados nodos, que están interconectados y que representan los contenidos de las páginas web y las relaciones entre ellos. Por su aspecto, la unión de todos los nodos se llama "árbol de nodos".

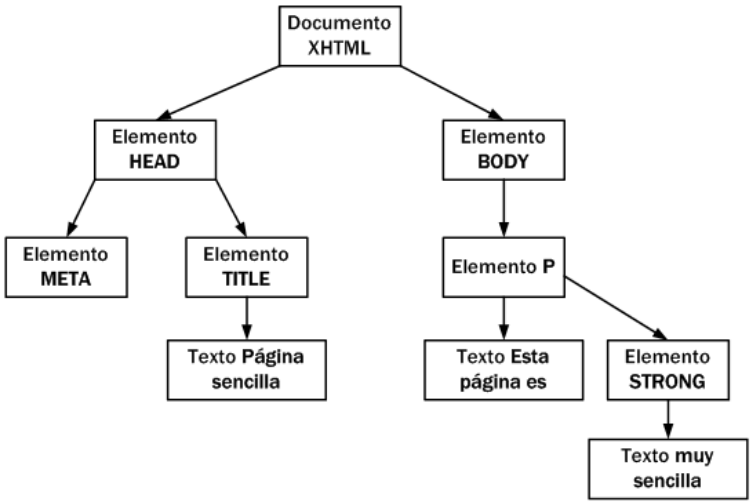



Figura 2.8. Árbol de nodos generado automáticamente por DOM a partir del código XHTML de la página

En el esquema anterior, cada rectángulo representa un nodo DOM y las flechas indican las relaciones entre nodos. Dentro de cada nodo, se ha incluido su tipo (que se verá más adelante) y su contenido.

La raíz del árbol de nodos de cualquier página XHTML siempre es la misma: un nodo de tipo especial denominado "Documento".

A partir de ese nodo raíz, cada etiqueta XHTML se transforma en un nodo de tipo "Elemento". La conversión de etiquetas en nodos se realiza de forma jerárquica. De esta forma, del nodo raíz solamente pueden derivar los nodos HEAD y BODY. A partir de esta derivación inicial, cada etiqueta XHTML se transforma en un nodo que deriva del nodo correspondiente a su "etiqueta padre".

La transformación de las etiquetas XHTML habituales genera dos nodos: el primero es el nodo de tipo "Elemento" (correspondiente a la propia etiqueta XHTML) y el segundo es un nodo de tipo "Texto" que contiene el texto encerrado por esa etiqueta XHTML.

Así, la siguiente etiqueta XHTML:

<title>Página sencilla</title>

Genera los siguientes dos nodos:


Figura 2.9. Nodos generados automáticamente por DOM para una etiqueta XHTML sencilla

De la misma forma, la siguiente etiqueta XHTML:

Esta página es muy sencilla

Genera los siguientes nodos:

- Nodo de tipo "Elemento" correspondiente a la etiqueta .
- Nodo de tipo "Texto" con el contenido textual de la etiqueta .
- Como el contenido de  incluye en su interior otra etiqueta XHTML, la etiqueta interior se transforma en un nodo de tipo "Elemento" que representa la etiqueta  y que deriva del nodo anterior.
- El contenido de la etiqueta  genera a su vez otro nodo de tipo "Texto" que deriva del nodo generado por .

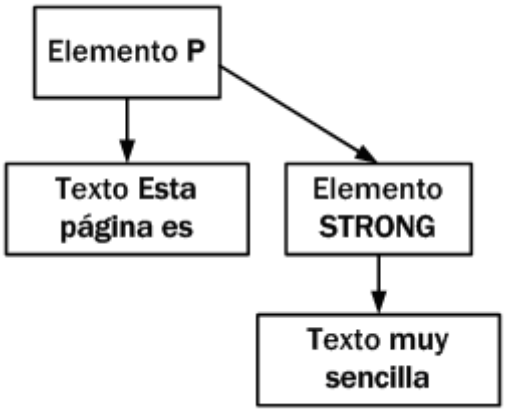



Figura 2.10. Nodos generados automáticamente por DOM para una etiqueta XHTML con otras etiquetas XHTML en su interior

La transformación automática de la página en un árbol de nodos siempre sigue las mismas reglas:

- Las etiquetas XHTML se transforman en dos nodos: el primero es la propia etiqueta y el segundo nodo es hijo del primero y consiste en el contenido textual de la etiqueta.

- Si una etiqueta XHTML se encuentra dentro de otra, se sigue el mismo procedimiento anterior, pero los nodos generados serán nodos hijo de su etiqueta padre.
- Como se puede suponer, las páginas XHTML habituales producen árboles con miles de nodos. Aun así, el proceso de transformación es rápido y automático, siendo las funciones proporcionadas por DOM (que se verán más adelante) las únicas que permiten acceder a cualquier nodo de la página de forma sencilla e inmediata.

#### 2.4.2.2. Tipos de Nodos.

La especificación completa de DOM define 12 tipos de nodos, aunque las páginas XHTML habituales se pueden manipular manejando solamente cuatro o cinco tipos de nodos:

- *Document,* nodo raíz del que derivan todos los demás nodos del árbol.
- *Element*, representa cada una de las etiquetas XHTML. Se trata del único nodo que puede contener atributos y el único del que pueden derivar otros nodos.
- *Attr,* se define un nodo de este tipo para representar cada uno de los atributos de las etiquetas XHTML, es decir, uno por cada par atributo=valor.
- *Text,* nodo que contiene el texto encerrado por una etiqueta XHTML.
- *Comment,* representa los comentarios incluidos en la página XHTML.

Los otros tipos de nodos existentes que no se van a considerar son *DocumentType, CDataSection, DocumentFragment, Entity, EntityReference, ProcessingInstruction y Notation*.

Una vez construido automáticamente el árbol completo de nodos DOM, ya es posible utilizar las funciones DOM para acceder de forma directa a cualquier nodo del árbol. Como acceder a un nodo del árbol es equivalente a acceder a *"un trozo"* de la página, una vez construido el árbol, ya es posible manipular de forma sencilla la página: acceder al valor de un elemento, establecer el valor de un elemento, mover un elemento de la página, crear y añadir nuevos elementos, etc.

DOM proporciona dos métodos alternativos para acceder a un nodo específico: acceso a través de sus nodos padre y acceso directo.

Las funciones que proporciona DOM para acceder a un nodo a través de sus nodos padre consisten en acceder al nodo raíz de la página y después a sus nodos hijos y a los nodos hijos de esos hijos y así sucesivamente hasta el último nodo de la rama terminada por el nodo buscado. Sin embargo, cuando se quiere acceder a un nodo específico, es mucho más rápido acceder directamente a ese nodo y no llegar hasta él descendiendo a través de todos sus nodos padre.

Por último, es importante recordar que el acceso a los nodos, su modificación y su eliminación solamente es posible cuando el árbol DOM ha sido construido

completamente, es decir, después de que la página XHTML se cargue por completo.

### 2.4.3. Java Script.

JavaScript [Eguíluz, 09] es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas.
Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario.

Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios.

A pesar de su nombre, JavaScript no guarda ninguna relación directa con el lenguaje de programación Java. Legalmente, JavaScript es una marca registrada de la empresa Sun Microsystems.

La sintaxis de un lenguaje de programación se define como el conjunto de reglas que deben seguirse al escribir el código fuente de los programas para considerarse como correctos para ese lenguaje de programación.

La sintaxis de JavaScript es muy similar a la de otros lenguajes de programación como Java y C. Las normas básicas que definen la sintaxis de JavaScript son las siguientes:

- No se tienen en cuenta los espacios en blanco y las nuevas líneas: como sucede con XHTML, el intérprete de JavaScript ignora cualquier espacio en blanco sobrante, por lo que el código se puede ordenar de forma adecuada para entenderlo mejor (tabulando las líneas, añadiendo espacios, creando nuevas líneas, etc.).
- Se distinguen las mayúsculas y minúsculas: al igual que sucede con la sintaxis de las etiquetas y elementos XHTML. Sin embargo, si en una página XHTML se utilizan indistintamente mayúsculas y minúsculas, la página se visualiza correctamente, siendo el único problema la no validación de la página. En cambio, si en JavaScript se intercambian mayúsculas y minúsculas el script no funciona.
- No se define el tipo de las variables: al crear una variable, no es necesario indicar el tipo de dato que almacenará. De esta forma, una misma variable puede almacenar diferentes tipos de datos durante la ejecución del script.
- No es necesario terminar cada sentencia con el carácter de punto y coma (;): en la mayoría de lenguajes de programación, es obligatorio terminar cada sentencia con el carácter; Aunque JavaScript no obliga a hacerlo, es conveniente seguir la tradición de terminar cada sentencia con el carácter del punto y coma (;).

- Se pueden incluir comentarios: los comentarios se utilizan para añadir información en el código fuente del programa.

La inigualable popularidad de JavaScript como lenguaje de programación de aplicaciones web se ha extendido a otras aplicaciones y otros entornos no relacionados con la web. Herramientas como Adobe Acrobat permiten incluir código JavaScript en archivos PDF.

Otras herramientas de Adobe como Flash y Flex utilizan ActionScript, un dialecto del mismo estándar de JavaScript. Photoshop permite realizar pequeños scripts mediante JavaScript y la versión 6 de Java incluye un nuevo paquete (denominado javax.script) que permite integrar ambos lenguajes. Por último, aplicaciones como Yahoo Widgets y el Dashboard de Apple utilizan JavaScript para programar sus widgets.

## 3. Metodología de SISAAC.

La finalidad del presente capitulo es describir los procedimientos y herramientas que se emplean para el desarrollo de SISAAC.

### 3.1. Ingeniería de Software.

La Ingeniería del Software [Pressman, 02] es una disciplina o área de la Informática o Ciencias de la Computación, que ofrece métodos y técnicas para desarrollar y mantener software de calidad que resuelven problemas de todo tipo.

#### 3.1.1. Definiciones.

Un actor: es una idealización de una persona, de un proceso, o de una cosa que interactúa con un sistema, un subsistema, o una clase. Un actor caracteriza las interacciones que los usuarios exteriores pueden tener con el sistema.

Un rol de clasificador, o simplemente “rol”, es la descripción de un objeto, que desempeña un determinado papel dentro de una interacción, distinta de los otros objetos de la misma clase.

#### 3.1.2. Etapas del ciclo de vida.

Al igual que en otros sistemas de ingeniería, los sistemas de software requieren un tiempo y esfuerzo considerable para su desarrollo y deben permanecer en uso por un periodo mucho mayor. Durante este tiempo de desarrollo y uso, desde que se detecta la necesidad de construir un sistema de software hasta que este es retirado, se identifican varias etapas que en conjunto se denominan el ciclo de vida del software y en cada caso, en función de cuales sean las características del proyecto, se configurará el ciclo de vida de forma diferente. Usualmente se consideran las etapas: especificación y análisis de requisitos, diseño del sistema, implementación del software, aplicación y pruebas, entrega y mantenimiento. Un aspecto esencial dentro de las tareas del desarrollo del software es la documentación de todos los elementos y especificaciones en cada fase. Dado que esta tarea siempre estará influida por la fase del desarrollo en curso, se explicará de forma distribuida a lo largo de las diferentes fases como un apartado especial para recalcar su importancia en el conjunto del desarrollo del software.

Las etapas principales a realizar en cualquier ciclo de vida son:

- Análisis: Construye un modelo de los requisitos
- Diseño: A partir del modelo de análisis se deducen las estructuras de datos, la estructura en la que descompone el sistema y la interfaz de usuario.
- Codificación: Construye el sistema. La salida de esta fase es código ejecutable.
- Pruebas: Se comprueba que se cumplen criterios de corrección y calidad.

- Mantenimiento: En esta fase, que tiene lugar después de la entrega se asegura que el sistema siga funcionando y adaptándose a nuevos requisitos.

### 3.1.3. Modelos.

Un modelo [Sommerville, 05] es una representación abstracta de un proceso del software. Cada modelo de proceso representa un proceso desde una perspectiva particular, y así proporciona sólo información parcial sobre ese proceso.

Modelo de Cascada: Separar en distintas fases de especificación y desarrollo.
Desarrollo Evolutivo: La especificación y el desarrollo están intercalados.
Prototipado: Un modelo sirve de prototipo para la construcción del sistema final.
Espiral: Cada ciclo en la espiral representa una fase del proceso del software.

#### 3.1.3.1. Modelo en Cascada.

El modelo en cascada solo se debe utilizar cuando los requerimientos se comprendan bien y sea improbable que cambien radicalmente durante el desarrollo del sistema.

Las ventajas de este modelo son que la documentación se produce en cada fase y que esté cuadra con otros modelos del proceso de ingeniería, su principal problema es su inflexibilidad al divisor el proyecto en distintas etapas. Se deben hacer compromisos en las etapas iníciales, lo que hace difícil responder a los cambios en los requerimientos del cliente.

Las Fases del modelo en cascada (ver Figura 3.1) se explican en las siguientes secciones.

##### 3.1.3.1.1. Análisis de requerimientos y definición.

Los servicios, restricciones y metas del sistema de definen a partir de las consultas con los usuarios. Entonces se definen detalles y sirven como una especificación del sistema.

##### 3.1.3.1.2. Diseño del sistema y del software.

El proceso de diseño del sistema divide los requerimientos en sistemas hardware o software. Establecer una arquitectura completa del sistema. El diseño del software identifica y describe las abstracciones fundamentales del sistema software y sus relaciones.

### 3.1.3.1.3. Implementación y prueba de unidades.

Durante esta etapa, el diseño del software se lleva a cabo como un conjunto o unidades de programas. La prueba de unidades implica verificar que cada una cumpla su especificación.

### 3.1.3.1.4. Integración y prueba del sistema.

Los programas o las unidades individuales de programa se integran y prueban como un sistema completo para asegurar que se cumplan los requerimientos del software. Después de las pruebas, el sistema software se entrega al cliente.

### 3.1.3.1.5. Funcionamiento y mantenimiento.

Por lo general (aunque no necesariamente), ésta es la fase más larga del ciclo de vida. El sistema se instala y se pone en funcionamiento práctico. El mantenimiento implica corregir errores no descubiertos en las etapas anteriores del ciclo de vida, mejorar la implementación de las unidades del sistema y resaltar los servicios del sistema una vez que se descubren nuevos requerimientos.
La dificultad en este modelo reside, en la dificultad de hacer cambios entre etapas. Debido al costo de producción y aprobación de documentos, las iteraciones son costosas e implican rehacer el trabajo. Por lo tanto, después de un número reducido de iteraciones, es normal congelar partes del desarrollo, como la especificación y continuar con las siguientes etapas de desarrollo.

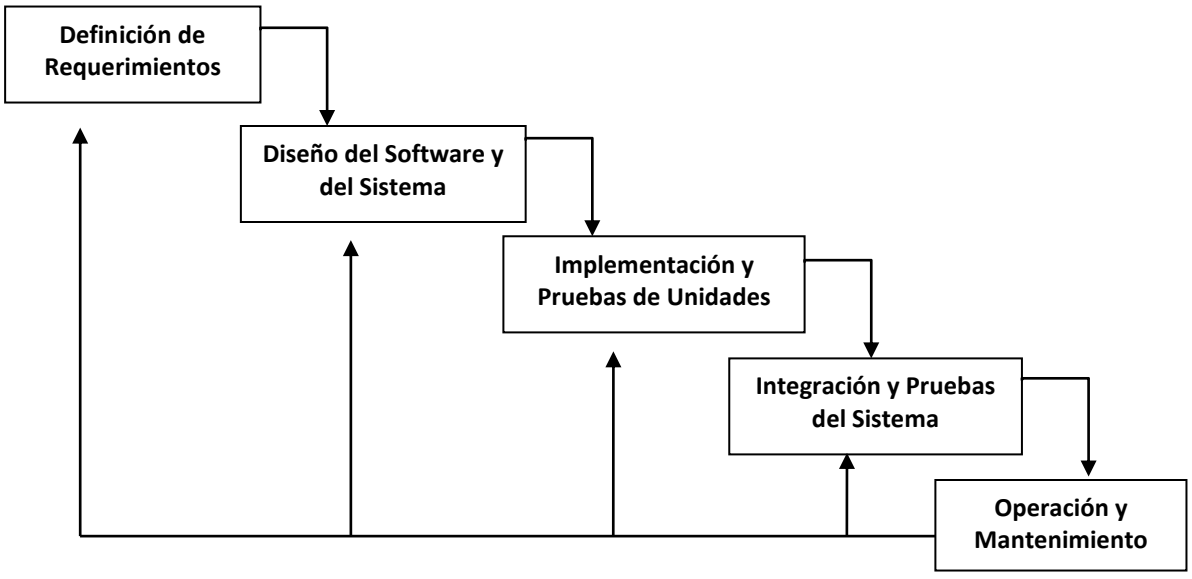



Figura 3.1 Fases del modelo en cascada.

### 3.1.3.2. Desarrollo evolutivo.

El desarrollo evolutivo (ver Figura 3.2) se basa en la idea de desarrollar una implementación inicial, exponiéndola a los comentarios del usuario y refinándola a través de las diferentes versiones hasta que se desarrolla un sistema adecuado, las actividades de especificación, desarrollo y validación se entrelazan en vez de separarse, con una rápida retroalimentación entre éstas.

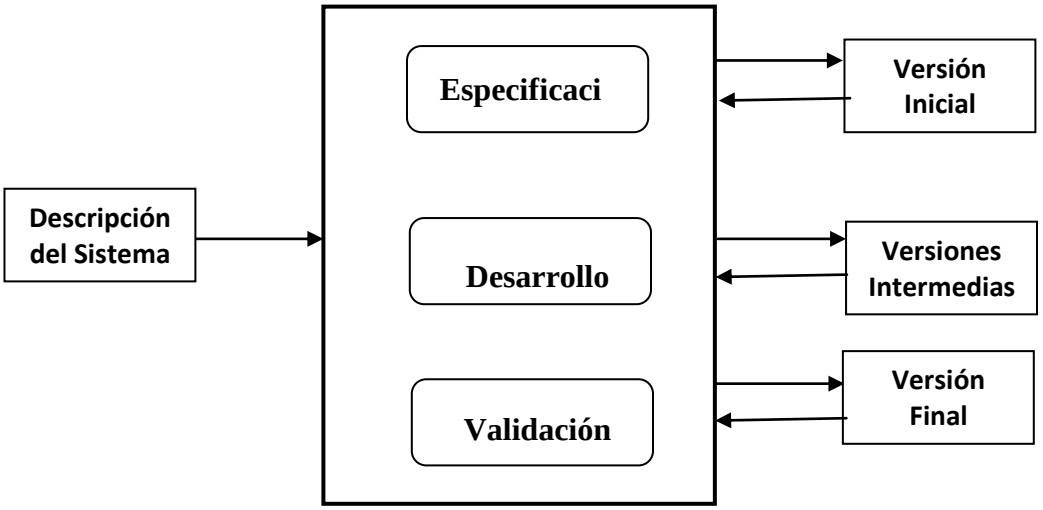



Figura 3.2 Esquema general del modelo de desarrollo evolutivo.

Existen dos tipos de desarrollo:

1. Desarrollo exploratorio, donde el objetivo del proceso es trabajar con el cliente para explorar sus requerimientos y entregar un sistema final. El desarrollo empieza con las partes del sistema que se comprende mejor. El sistema evoluciona agregado nuevos atributos propuestos por el cliente.

2. Prototipos desechables, Donde el objetivo del proceso de desarrollo evolutivo es comprender los requerimientos del cliente y entonces desarrollar una definición mejorada de los requerimientos para el sistema. El prototipo se centra en experimentar con los requerimientos del cliente que no se comprenden del todo.

En la producción de sistema, un enfoque evolutivo para desarrollo de software suele ser más efectivo que el enfoque en cascada ya que satisface las necesidades inmediatas de los clientes. Los problemas de este modelo son:

1. El proceso no es visible. Los administradores tienen que hacer entregas regulares para medir el proceso. Si los sistemas se desarrollan rápidamente, no es rentable producir documentos que reflejan cada versión del sistema.

2. A menudo los sistemas tienen una estructura deficiente: los cambios continuos tienden a corromper la estructura del software. Incorpora cambios en él se convierte cada vez más en una tarea difícil y costosa.

Este modelo es aplicable para:

- Sistemas interactivos pequeños o medianos.
- Partes de sistemas grandes (p.ej. la interfaz de usuario).
- Sistemas de corta vida.

### 3.1.3.3. Prototipado.

El modelo de prototipos se utiliza para dar al usuario una vista preliminar de parte del software. Este modelo es básicamente prueba y error, ya que si al usuario no le gusta una parte del prototipo significa que la prueba fallo por lo cual se debe corregir el error que se tenga hasta que el usuario quede satisfecho.
Además, el prototipo debe ser construido en poco tiempo, usando los programas adecuados y no se debe utilizar mucho dinero, pues a partir de que éste se pruebe se puede iniciar el verdadero desarrollo del software. Pero eso sí, al construir el prototipo se asegura que el software sea de mejor calidad, además de que su interfaz sea de agrado para el usuario. Un prototipo podrá ser construido solo sí es posible experimentar con el software.
Sus desventajas son que debido a que el usuario ve que el prototipo funciona piensa que este es el producto terminado y no entienden que recién se va a desarrollar el software. Otro problema es que el prototipo deber ir acompañado de otro modelo pasa su desarrollo. Existen dos tipos de prototipado:

- Prototipado exploratorio: El objetivo es trabajar con clientes hasta evolucionar a un sistema final, a partir de una especificación inicial. Se debe comenzar con unas especificaciones bien entendidas.
- Prototipado de *"throw-away"* (usar y tirar): El objetivo es entender los requerimientos del sistema. Se puede comenzar con especificaciones poco entendidas.

### 3.1.3.4. Modelo de proceso de espiral .

El modelo en espiral del proceso del software (ver Figura 3.3) fue propuesto por Boehm. Más que representar el proceso del software como una secuencia de actividades con retrospectiva de una actividad a otra, se representa como un espiral. Cada ciclo en la espiral representa una fase del proceso del software. Así el ciclo más interno podría referirse a la viabilidad del sistema, el siguiente ciclo a la definición de requerimientos, el siguiente ciclo al diseño del sistema, y así sucesivamente.

Cada ciclo de la espiral se divide en cuatro sectores:

1. Definiciones de objetivos. Para esta fase del proyecto se definen los objetivos específicos. Se identifican las restricciones del proceso y el

producto, y se traza un plan detallado de gestión. Se identifican los riesgos del proyecto. Dependiendo de estos riesgos, se planean estrategias alternativas.

2. Evolución y reducción de riesgos. Se lleva a cabo un análisis detallado para cada uno de los riesgos del proyecto identificados. Se definen los pasos para reducir dichos riesgos. Por ejemplo, si existe el riesgo de tener requerimientos inapropiados, se puede desarrollar un prototipo del sistema.
3. Desarrollo y validación. Depende de la evaluación de riesgos, se elige un modelo para el desarrollo del sistema. Por ejemplo, si los riesgos en la interfaz del usuario son dominantes, un modelo de desarrollo apropiado podría ser la construcción de prototipos evolutivos. Si los riesgos de seguridad son principal consideración, un desarrollo basado en transformaciones formales podría ser más apropiado, y así sucesivamente. El modelo en cascada podría ser más apropiado para el desarrollo si el mayor riesgo identificado es la integración de los subsistemas.
4. Planificación. El proyecto se revisa y se toma la decisión de si debe continuar con un ciclo posterior de la espiral. Si se decide continuar, se desarrollan los planes para la siguiente fase del proyecto.

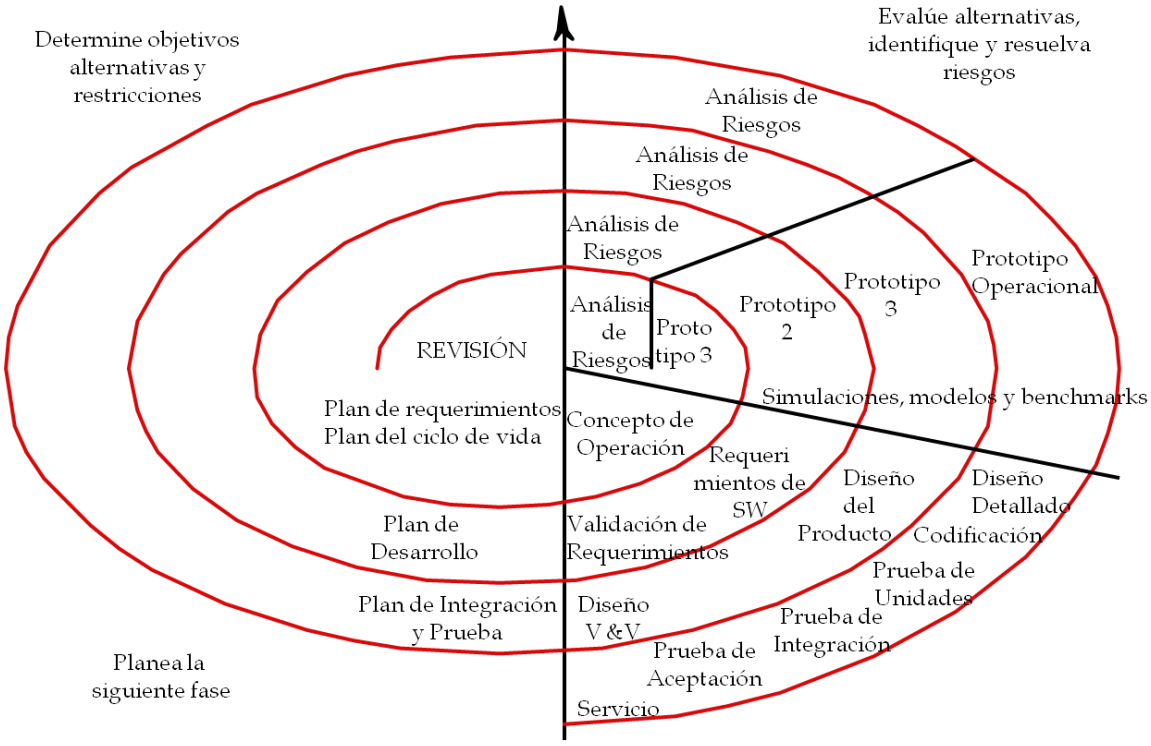



Figura 3.3 Esquema general del modelo en espiral

Las principales ventajas del modelo son:

- Centra su atención en la reutilización de componentes y eliminación de errores en información descubierta en fases iníciales.
- Los objetivos de calidad son el primer objetivo.

- Integra desarrollo con mantenimiento.
- Provee un marco de desarrollo de hardware/software.

Problemas que presenta el Modelo en Espiral:

- El desarrollo contractual especifica el modelo del proceso y los resultados a entregar por adelantado.
- Requiere de experiencia en la identificación de riesgos.
- Requiere refinamiento para uso generalizado.

### 3.1.4. ADOO.

Análisis y Diseño Orientado a Objetos (ADOO) es un enfoque de la ingeniería de software que modela un sistema como un grupo de objetos que interactúan entre sí. Este enfoque representa un dominio en términos de conceptos compuestos por verbos y sustantivos, clasificados de acuerdo a su dependencia funcional.

En este método de análisis y diseño se crea un conjunto de modelos utilizando una notación acordada como, por ejemplo, el lenguaje unificado de modelado (UML). ADOO aplica técnicas de modelado de objetos para analizar los requerimientos para un contexto - por ejemplo, un sistema de negocio, un conjunto de módulos de software - y para diseñar una solución para mejorar los procesos involucrados. No está restringido al diseño de programas de computadora, sino que cubre sistemas enteros de distinto tipo. Las metodologías de análisis y diseño más modernas son casos de uso guiados a través de requerimientos, diseño, implementación, pruebas, y despliegue. El lenguaje unificado de modelado se ha vuelto el lenguaje de modelado estándar usado en análisis y diseño orientado a objetos.

### 3.1.5. UML.

UML Lenguaje Unificado de Modelado [Rumbaugh, 00] es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Captura decisiones y conocimiento sobre los sistemas que se deben construir. Se usa para entender, diseñar, hojear configurar, mantener y controlar la información sobre tales sistemas. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios.
El lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorpora las mejores prácticas actuales en un acercamiento estándar. UML incluye conceptos semánticos, natación y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas.
UML proporciona un vocabulario que incluye tres categorías: Elementos, Diagramas y Relaciones (ver Figura 3.4), que serán descritos en las siguientes secciones.

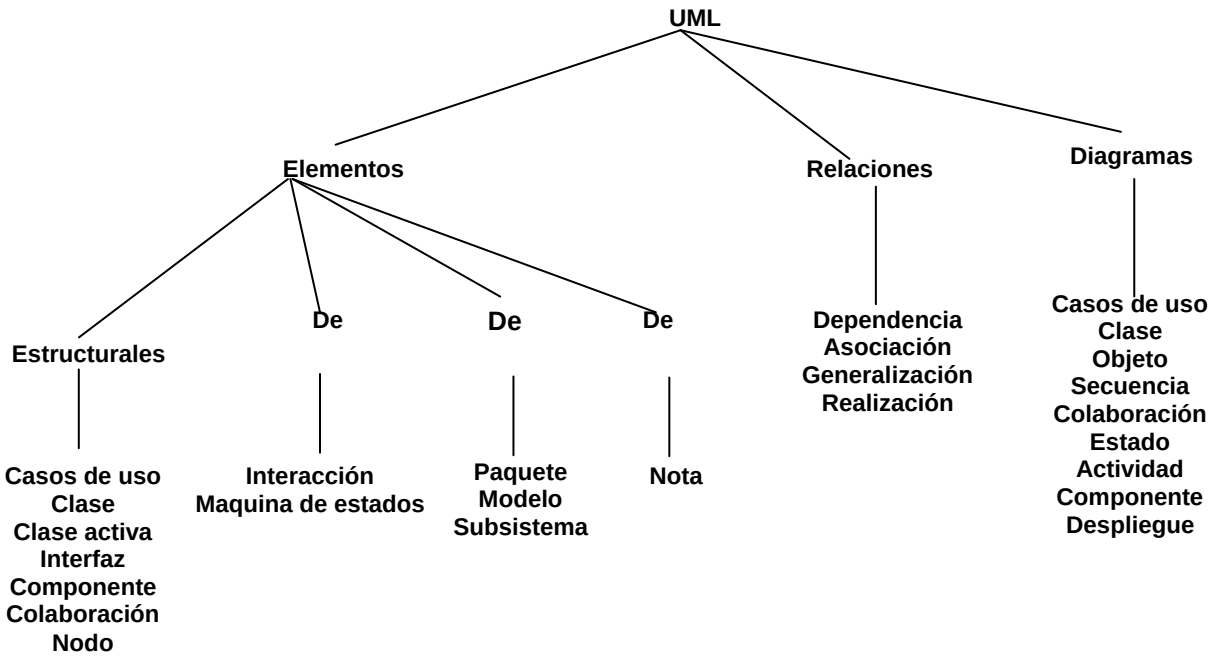



Figura 3.4 Elementos de UML

### 3.1.5.1. Elementos.

Los elementos En UML son las abstracciones de primer Nivel, los elementos de UML se clasifican en 4 tipos, estructurales, de comportamiento, de agrupación y de anotación.

1. Estructurales: Describe los elementos del sistema y sus relaciones con otros elementos. Son las partes estáticas del modelo y representan cosas que son conceptuales o materiales. Proporcionan la base sobre la cual se construye el comportamiento dinámico.
2. De comportamiento: El comportamiento describe el comportamiento dinámico de un sistema en el tiempo y espacio.
3. De agrupación: Forman la parte organizativa de los elementos UML. El principal elemento de agrupación paquete, que es una unidad de organización de uso general para poseer y manejar el contenido de modelo. Cada elemento está contenido en algún paquete.
4. De anotación: Son las partes explicativas de los modelos UML. Son comentarios que se pueden aplicar para describir, clasificar y hacer observaciones sobre cualquier elemento de un modelo.
   El tipo principal de anotación es la nota que simplemente es un símbolo para mostrar restricciones y comentarios junto a un elemento o un conjunto de elementos.

**Hay siete tipos de *elementos estructurales:* casos de uso, clases, clases activas, interfaces, componentes, colaboraciones y nodos.**

**Casos de Uso:** Un caso de uso en una unidad de funcionalidad, extremadamente visible, proporciona una unidad del sistema y expresada por secuencias de mensajes intercambiados por la unidad del sistema y uno o más actores, se representa con una elipse con borde continuo (ver Tabla 3.1)

**Clases**: son una descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semántica. Una clase implementa una o más interfaces. Gráficamente se representa como un rectángulo que incluye su nombre, sus atributos y sus operaciones (ver Tabla 3.1).

**Clases Activas:** son las clases cuyos objetos tienen uno o más procesos o hilos de ejecución por lo y tanto pueden dar lugar a actividades de control. Una clase activa es igual que una clase, excepto que sus objetos representan elementos cuyo comportamiento es concurrente con otros elementos. Se representa igual que una clase, pero con líneas más gruesas (ver Tabla 3.1).

**Interfaces:** son la descripción del comportamiento de objetos sin dar su implementación o estado; una interfaz contiene operaciones pero no atributo, y no tiene asociaciones salientes que muestren la visibilidad desde la propia interfaz.
Una o más clases o componentes pueden realizar una interfaz, y cada clase implementa las operaciones de la interfaz. (Ver Tabla 3.1)

**Componentes:** Un componente es una parte física y reemplazable de un sistema que conforma con un conjunto de interfaces y proporciona la implementación de dicho conjunto. Un componente representa típicamente el empaquetamiento físico de diferentes elementos lógicos, como clases, interfaces y colaboraciones. En la Tabla 3.1 se muestra la representación su representación grafica.

**Colaboraciones:** Definen una interacción y es una sociedad de **roles** y otros elementos que colaboran para proporcionar un comportamiento cooperativo mayor que la suma de los comportamientos de sus elementos. Las colaboraciones tienen una dimensión tanto estructural como de comportamiento. Una misma clase puede participar en diferentes colaboraciones. Las colaboraciones representan la implementación de patrones que forman un sistema. Se representa mediante una elipse con borde discontinuo (ver Figura 3.1).

**Nodos:** Un nodo es un elemento físico que existe en tiempo de ejecución y representa un recurso computacional que, por lo general, dispone de algo de memoria y, con frecuencia, de capacidad de procesamiento. Un conjunto de componentes puede residir en un nodo (ver Tabla 3.1).

**Hay dos tipos de *elementos de comportamiento:* interacciones y máquinas de estados.**

**Interacción:** describe secuencias de intercambio de mensajes entre los roles que implementan el comportamiento de un sistema, es decir, es un comportamiento que comprende un conjunto de mensajes intercambiados entre un conjunto de objetos, dentro de un contexto particular para conseguir un propósito específico. Una interacción involucra otros muchos elementos, incluyendo mensajes, secuencias de acción (comportamiento invocado por un objeto) y enlaces (conexiones entre objetos).
Este visón proporciona una vista integral del comportamiento de un sistema, es decir, muestra el flujo de control a través de muchos objetos.
La representación de un mensaje es una flecha dirigida que normalmente con el nombre de la operación (ver Tabla 3.2).

**Maquina de estado:** Es un comportamiento que especifica las secuencias de estados por las que van pasando los objetos o las interacciones durante su vida en respuesta a eventos, junto con las respuestas a esos eventos. Una maquina de estados involucra otros elementos como son estados, transiciones (flujo de un estado a otro), eventos (que disparan una transición) y actividades (respuesta de una transición) (ver Tabla 3.2).

**Hay cuatro *tipos de agrupaciones:* paquetes, modelos, subsistemas y marcos de trabajo.**

**Paquetes:** son unidades para manipular el contenido de un modelo, así como unidades para el control de acceso y el control de configuración. Cada elemento del modelo pertenece a un paquete o a otro elemento, su representación Grafica se muestra en la Tabla 3.3.

**Modelos:** son una descripción completa de un sistema, con una determinada precisión desde un punto de vista. Puede haber varios modelos de un sistema desde distintos puntos de vista. Por ejemplo, un modelo de análisis y un modelo de diseño. Un modelo se representa como un a clase especial de paquete.

**Subsistemas:** es otro paquete especial. Representa una porción de un sistema, con una interfaz perfectamente determinada, que puede ser implementado como un componente distinto, la representación grafica se encuentra en la Tabla 3.3

**Hay un tipo de *elemento de anotación:* notas** Los elementos de anotación son las partes explicativas de los modelos UML. Son comentarios que se pueden aplicar para describir, clasificar y hacer observaciones sobre cualquier elemento de un modelo. El tipo principal de anotación es la **nota** que simplemente es un símbolo para mostrar restricciones y comentarios junto a un elemento o un conjunto de elementos.

### 3.1.5.2. Relaciones UML.

Existen cuatro tipos de relaciones entre los elementos de un modelo UML. Dependencia, asociación, generalización y realización, estas se describen a continuación:

1. Dependencia: Una dependencia indica una relación semántica entre dos o más elementos del modelo. Relaciona los elementos del modelo entre ellos, y se requiere un conjunto de instancias para su significado, su representación grafica se muestra en la Tabla 3.4.
2. Asociación: Una asociación describe conexiones discretas entre objetos u otras instancias de un sistema. Una asociación relaciona una lista ordenada (tuplas) de dos o más clasificadores, con las repeticiones permitidas. El tipo más común de asociaciones es una asociación binaria entre un par de clasificadores. Una instancia de asociación es un enlace. Un enlace abarca una tupla (una lista ordenada) de objetos, cada uno dibujado a partir de su clase correspondiente. Un enlace binario abarca un par de objetos. Las asociaciones llevan información sobre relaciones entre objetos en un sistema. Cuando se ejecuta un sistema, los enlaces entre objetos se crean y se destruyen. Las asociaciones son el "pegamento" que mantiene unido a un sistema. Sin asociaciones no hay nada más que clases aisladas que no trabajan juntas. Una asociación se representa gráficamente con una línea recta solida, (ver Tabla 3.4).
3. Generalización: Es una relación de especialización / generalización en la cual los objetos del elemento especializado (el hijo) pueden sustituir a los objetos del elemento general (el padre). De esta forma, el hijo comparte la estructura y el comportamiento del padre. Gráficamente, la generalización se representa con una línea con punta de flecha vacía,(ver Tabla 3.4).
4. Realización: Es una relación semántica entre clasificadores, donde un clasificador especifica un contrato que otro clasificador garantiza que cumplirá. Se pueden encontrar relaciones de realización en dos sitios: entre interfaces y las clases y componentes que las realizan, y entre los casos de uso y las colaboraciones que los realizan. La realización se representa como una mezcla entre la generalización y la dependencia, esto es, una línea discontinua con una punta de flecha vacía (ver Tabla 3.4).

### 3.1.5.3. Diagramas UML.

Los diagramas se utilizan para representar diferentes perspectivas de un sistema de forma que un diagrama es una proyección del mismo. UML proporciona un amplio conjunto de diagramas que normalmente se usan en pequeños subconjuntos para poder representar las cinco vistas principales de la arquitectura de un sistema.

### 3.1.5.3.1. Diagrama de clases.

Muestran un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Estos diagramas son los más comunes en el modelado de sistemas orientados a objetos y cubren la vista de diseño estática o la vista de procesos estática (sí incluyen clases activas) (ver Figura 3.5).

Es un modelo estático del sistema que contiene los siguientes elementos:

- Clases
- Interfaces
- Colaboraciones: Conjunto de clases, interfaces y demás que exhiben un comportamiento.
- Relaciones de dependencia, colaboración y asociación.

Se puede utilizar para modelar tres cosas:

1. Los elementos del sistema.
2. Colaboraciones.
3. Esquema lógico de bases de datos.

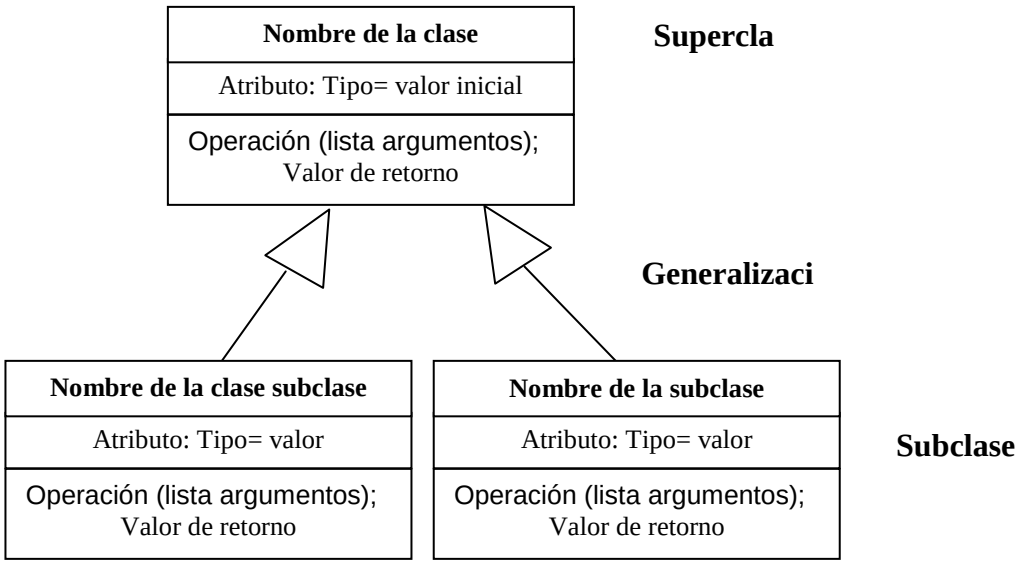



Figura 3.5: Diagrama de clases

### 3.1.5.3.2. Diagrama de Objetos.

Es un diagrama que contiene objetos y enlaces entre ellos. Pueden también agruparse en paquetes y se puede mostrar las clases si se considera oportuno para mostrar los objetos que vienen de una clase concreta. Sirve para tomar una

instantánea del sistema en un momento dado del funcionamiento. También es un modelo estático porque no representa la evolución a través del tiempo, sino un momento concreto (ver Figura 3.6).

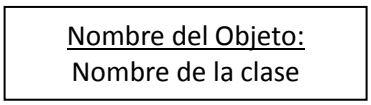



Figura 3.6: Diagrama de Objetos

### 3.1.5.3.3. Diagrama de Casos de uso.

Los casos de uso son una descripción de una interacción con el sistema por parte de algo externo al sistema que se llama actor. Por ejemplo: un usuario pulsa el botón de llamada de un ascensor. El nombre del caso de uso se pone en la elipse, Un caso de uso es un patrón o comportamiento que exhibe el sistema. Los casos de uso representan requisitos funcionales. Los casos de uso se escriben desde el punto de vista del actor, que es el usuario o sistema externo que interactúa con el sistema (ver Figura 3.7).

Relaciones que nos podemos encontrar en los casos de uso

1. *Include*: Es el concepto de subrutina. Si por ejemplo tenemos dos casos de uso A y B que tienen una serie de pasos en común se ponen esos pasos en un tercer caso de uso C y A y B lo incluyen para usarlo.
2. *Extends*: Significa que un caso de uso B es igual al A al cual extiende añadiendo algunos pasos.
3. *Comunicates*: Comunica un actor con un caso de uso o con otro actor.

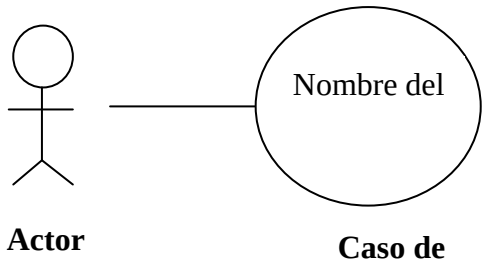



Figura 3.7: Diagrama de casos de Uso

### 3.1.5.3.4. Diagrama de Secuencias.

El diagrama de secuencia muestra un conjunto de mensajes, dispuestos en una secuencia temporal. Cada rol en la secuencia muestra como una línea de vida, es decir un alinea vertical que representa un rol durante cierto pazo de tiempo, con la interacción completa. Los mensajes se muestran como flechas entre las líneas de vida. Un diagrama de secuencias puede mostrar un escenario, es decir, una historia individual de una transacción.

Un uso de un diagrama de secuencia es montar la secuencia del comportamiento de un caso de uso. Cuando está implementado el comportamiento, cada mensaje en un diagrama de secuencia corresponde a una operación de una clase, aun evento disparador, o ha a una transacción en una máquina de estados.
Muestra los objetos y los mensajes intercambiados entre ellos teniendo en cuenta la temporalidad con la que ocurren. Pueden mostrarse también los componentes porque son objetos reutilizables y los casos de uso porque se muestran como objetos que implementan el caso de uso. Sirven para documentar el diseño y validar los casos de uso (ver Figura 3.8).

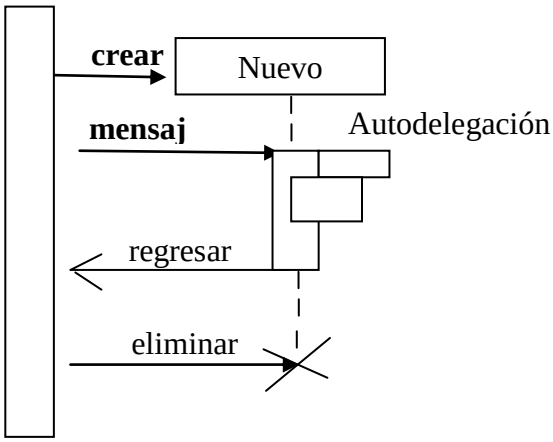



Figura 3.8: Diagrama de Secuencias

### 3.1.5.3.5. Diagrama de Colaboración.

Una colaboración modela los objetos y los enlaces significativos dentro de una interacción. Los objetos y los enlaces son significativos solamente en el contexto proporcionado por la interacción. Un rol describe un objeto, y un rol en la asociación describe un enlace dentro de una colaboración. Un diagrama de colaboración muestra los roles en la interacción en una disposición geométrica. Los mensajes se muestran como flechas, ligadas a la líneas de la relación, que conectan a los roles. La secuencia de mensajes, se indica con los números secuenciales que proceden a la descripción del mensaje.
Un uso de los diagramas de colaboración es mostrar la implementación de una operación. La colaboración muestra los parámetros y las variables locales de la operación, así como asociaciones más permanentes. Cuando se implementa el comportamiento, la secuencia de los mensajes corresponde a la estructura de llamadas anidadas y el paso de señales del programa (ver Figura 3.9).

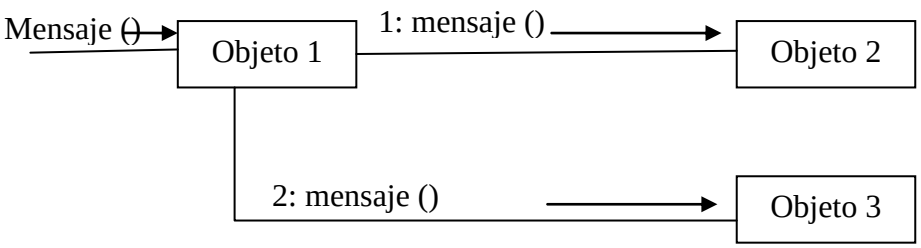





Figura 3.9: Diagrama de Colaboración

### 3.1.5.3.6. Diagrama de Distribución.

Refleja la organización del *hardware.* El elemento principal de este diagrama es el nodo. Hay dos tipos: los nodos con capacidad de ejecutar componentes y los que no pueden ejecutar. La información que hay en un nodo es: nombre del nodo y componentes del nodo (se puede poner sólo sus nombres o un diagrama de componentes). Los nodos a su vez pueden estar físicamente conectados con otros, lo que se representa con una línea que los une. La utilidad principal de este tipo de diagramas es la modelización de una red (ver Figura 3.10).

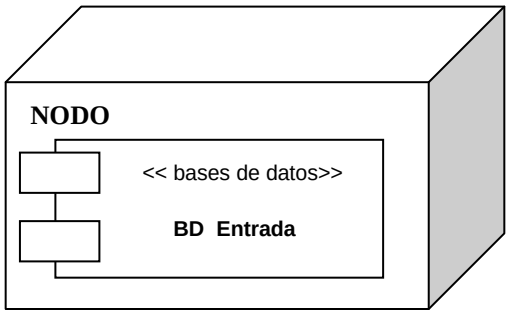



Figura 3.10: Diagrama de Distribución

### 3.1.5.3.7. Diagrama de Estados.

Muestran una maquina de estados compuesta por estados, transiciones, eventos y actividades. Estos diagramas cubren la vista dinámica de un sistema y son muy importantes a la hora de modelar el comportamiento de una interfaz, clase o colaboración (ver Figura 3.11).
Estado: Captura el estado de un objeto durante cierto periodo.

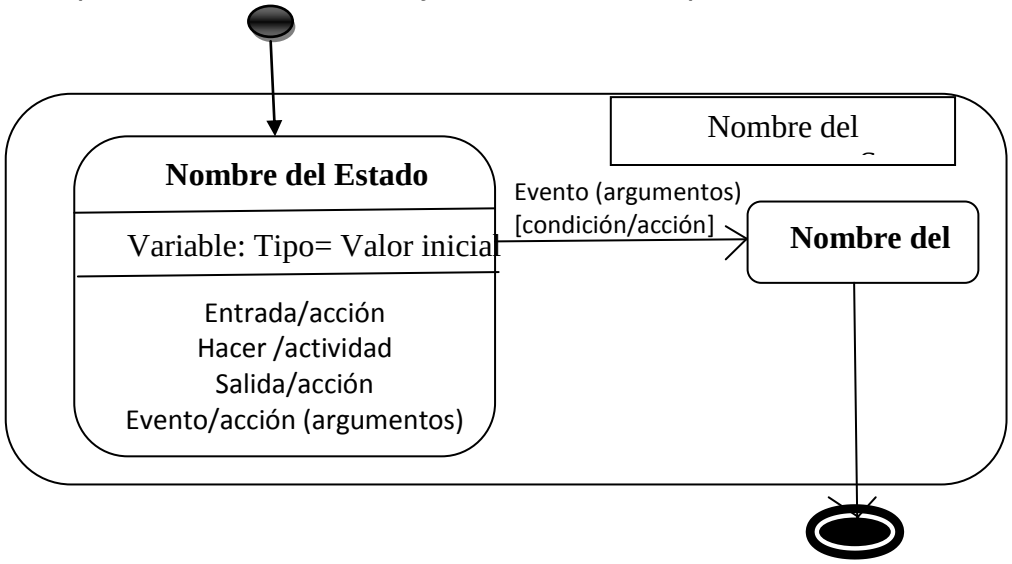



Figura 3.11: Diagrama de Estado

### 3.1.6.3.8. Diagrama de Actividades:

Un grafo de actividades o Diagrama de Actividades es una variante de una máquina de estados, que muestra las actividades de composición implicadas en la ejecución de un cálculo. Un estado de actividades representa una actividad: un paso en el flujo de trabajo o la ejecución de una operación. Un grafo de actividades describe grupos secuenciales y concurrentes de actividades. Los grafos de actividades se muestran en diagramas de actividades. Las flechas muestran dependencias secuenciales por ejemplo, las representaciones deben ser seleccionadas antes de poder antes de poder planificarlas. Las barras horizontales representan bifurcaciones o uniones de control. Los diagramas de actividades se pueden también utilizar para modelar actividades de software. Un diagrama de actividades es provecho para entender el comportamiento de alto nivel de la ejecución de un sistema, sin profundizar en los detalles internos de los mensajes, lo que requería un diagrama de colaboración.
Los parámetros de entrada y de salida de una acción se pueden mostrar usando las relaciones de flujos que conectan la acción y un estado de flujo del objeto (ver Figura 3.12).

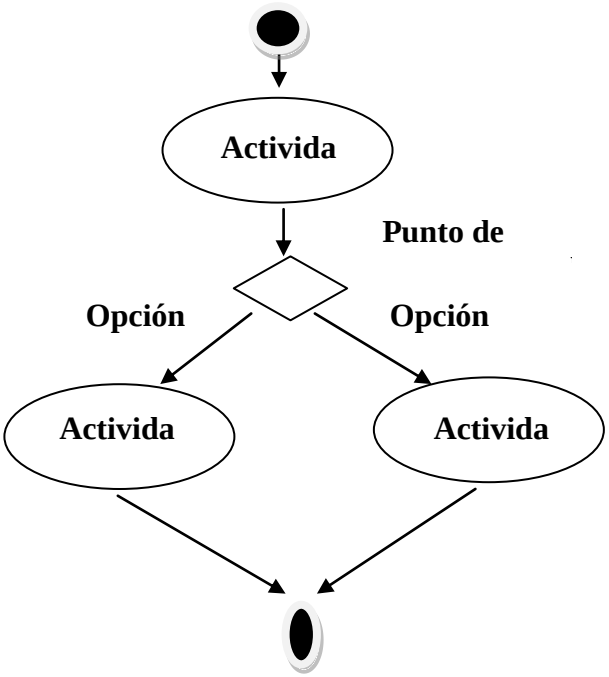



Figura 3.12: Diagrama de Actividades

### 3.1.5.3.9. Diagramas de Componentes.

Muestra la organización y las dependencias entre un conjunto de componentes. Cubren la vista de la implementación estática y se relacionan con los diagramas de clases ya que en un componente suele tener una o más clases, interfaces o colaboraciones (ver Figura 3.13).

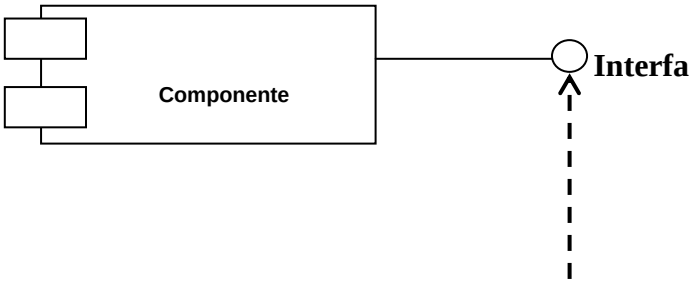
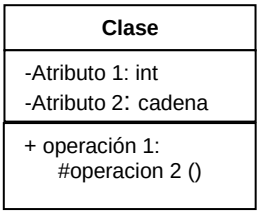
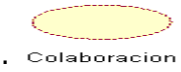
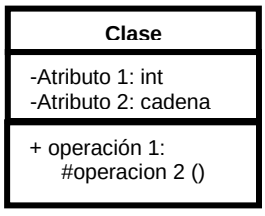
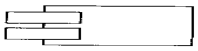


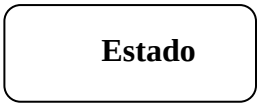

Figura 3.13: Diagrama de componentes

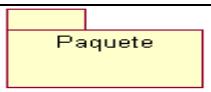
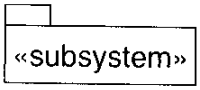
### 3.1.5.4. Notación Gráfica.

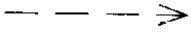
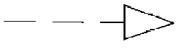
En las siguientes tablas se muestra la representación gráfica de las relaciones UML

**Tabla 3.1 Elementos estructurales**

| Clasificador | Función | Notación |
|---|---|---|
| Clase | Un concepto del sistema modelado |   |
| Interfaz | Un conjunto de operaciones con nombre que caracteriza un comportamiento |   |
| Colaboración | Define una interacción entre elementos que cooperan para proporcionar un comportamiento mayor que la suma de los comportamientos de sus elementos |   |
| Caso de uso | Una especificación del comportamiento de una identidad y su interacción con los agentes externos. |  |
| Clase activa | Se trata de una clase, en la que existen procesos o hilos de ejecución concurrentes con otros elementos. Las líneas del contorno son más gruesas que en la clase "normal" |   |
| Componente | Una pieza física de un sistema |  |
| Nodo | Un recurso computacional |  |

| Tabla 3.2 Elementos de comportamiento | | |
|---|---|---|
| **Clasificador** | **Función** | **Notación** |
| Interacción | Comprende un conjunto de mensajes que se intercambian entre un conjunto de objetos, para cumplir un objetivo especifico. |   |
| Máquinas de estados | Especifica la secuencia de estados por los que pasa un objeto o una interacción, en respuesta a eventos. |   |

| Tabla 3.3 Elementos Agrupación | | |
|---|---|---|
| **Clasificador** | **Función** | **Notación** |
| Paquete | Se emplea para organizar otros elementos en grupos. |   |
| Subsistema | un paquete que es tratado como una unidad con una especificación, implementación, e identidad. |   |

| Tabla 3.4 Tipos de relaciones | | |
|---|---|---|
| **Relación** | **Función** | **Notación** |
| Asociación | Una descripción de una conexión entre instancias de clases. |   |
| Dependencia | Una relación entre dos elementos del modelo. |  |
| Flujo | Una relación entre dos versiones de un objeto en sucesivas veces. |  |
| Generalización | Una relación entre una descripción mas general y una variedad más específica de la general, usada para herencia. |  |
| Realización | Relación entre una especificación y su implementación. |  |

## 3.2. Análisis y Diseño del Entorno SISAAC.

En las secciones posteriores se describen las etapas del Análisis y Diseño e Implementación del sistema SISAAC es importante hacer mención que se siguió el ciclo de vida en espiral (Ver Capitulo 3)**.** Se establecieron lo objetivos específicos, sus restricciones del sistema, y se planeo el desarrollo de acuerdo a los objetivos y restricciones, se desarrollaron prototipos del sistema para identificar si existiera algún riesgo de tener requerimientos inapropiados.

En cada etapa del desarrollo de la plataforma se hizo la revisión y de acuerdo a ello se decide si se debe continuar a la siguiente fase del desarrollo.

### 3.2.1. Análisis.

De acuerdo al proceso de desarrollo se tiene el análisis del sistema, por lo que las siguientes secciones corresponden al presente tema.

#### 3.2.1.2. Introducción.

Al realizar el análisis del comportamiento general del sistema denominado SISAAC podemos listar una serie de requisitos divididos de acuerdo al tipo de usuario, los cuales pueden ser: administrador y usuario.

Por el lado del administrador se tiene dos requisitos del sistema: Pre-registro y modificar y/o gestionar la BD Usuarios.

Por el lado del usuario se identificaron los siguientes requisitos: Registro en el sistema, Ingreso al sistema y unirse o crear grupos de trabajo.

Para el diseño de los casos de uso tomamos de base el listado de los requisitos de SISAAC anteriormente mencionados, obtenidos a partir de los posibles requerimientos del usuario y administrador.

En la sección siguiente se muestran los casos de uso del comportamiento general y de cada requisito del sistema SISAAC.

#### 3.2.1.2. Casos de uso.

En la presente sección mostramos los casos de uso de SISAAC.

Comenzamos describiendo el caso de uso general de SISAAC, el cual muestra un esquema del comportamiento general de SISAAC (ver Figura 3.14), dónde se puede observar que participan dos actores (usuario y administrador). En la Tabla 3.5 se muestra el esquema general de casos de uso del funcionamiento de SISAAC.

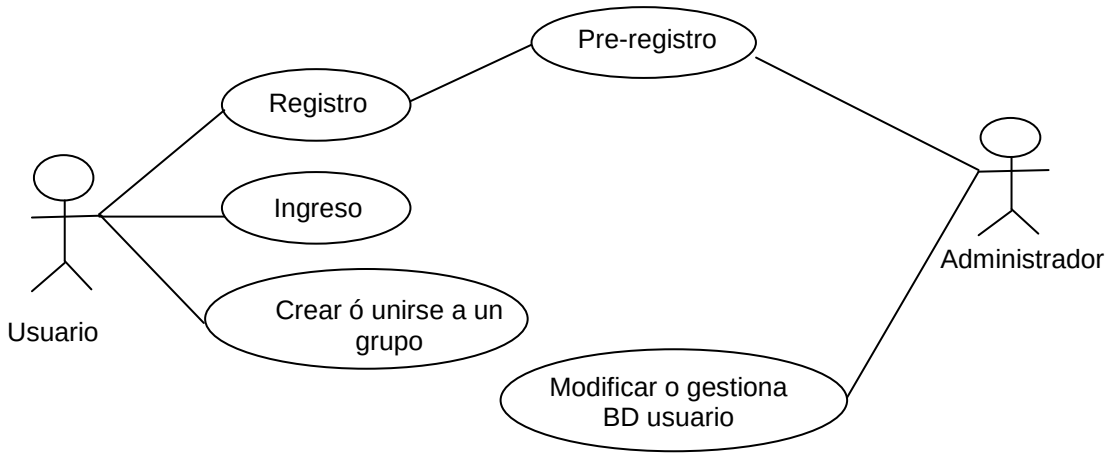



Figura 3.14 Casos de uso del comportamiento general de SISAAC.

| | |
|---|---|
| Nombre: | Esquema General de SISAAC |
| Descripción: | Funcionamiento de SISAAC |
| Actores: | Administrador, Usuario. |
| Precondiciones: | Estar registrado en la BD usuarios (ser alumno de la FCC) |
| Flujo Normal : | 1) Administrador realiza pre-registro<br>2) Administrador gestiona, administra la BD de los usuarios<br>3) El usuario se registra en el sistema<br>4) El usuario ingresa al sistema<br>5) El usuario puede crear o unirse a algún grupo existente<br>6) El usuario puede modificar su información personal. |
| Flojo Alternativo: | 3) Si el Usuario ingresa datos erróneos al sistema, éste mostrará un mensaje de error permitiéndole que vuelva a ingresar sus datos.<br>5) Si el usuario no ingresa caracteres o ingresa un número mayor a 255 caracteres no podrá crearse el grupo. |

Tabla 3.5 Detalles del Esquema general de casos de uso del Funcionamiento de SISAAC.

**Caso de uso Pre-registro** el actor es el administrador como lo podemos observar en la Figura 3.15 el cual realiza el pre-registro de los usuarios**.** En la Tabla 3.6 se muestra a detalle el caso de uso pre-registro.

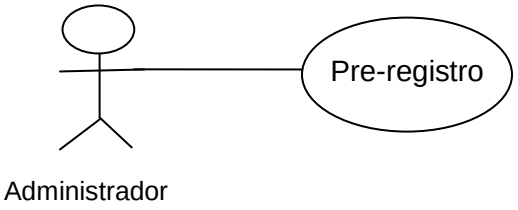
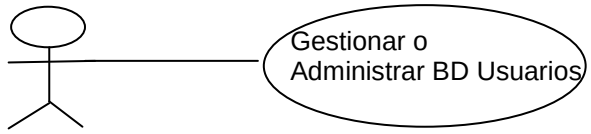



Figura 3.15 Caso de Uso: Pre-registro.

| | |
|---|---|
| Nombre: | Realizar Pre registro |
| Descripción: | El administrador cargará la información en la BD usuarios. |
| Actores: | Administrador |
| Precondiciones: | Ser Alumno de la FCC |
| Flujo Normal : | 1) El administrador ingresa la información del alumno de la FCC a la BD usuarios. |
| Flojo Alternativo: | 1) Si el usuario ya existe en la BD, el sistema Manda un mensaje que ya ha sido cargado anteriormente. |

Tabla 3.6 Detalle del caso de Uso de Pre registro.

**Caso de uso: Gestionar o Administrar BD Usuarios** (Ver Figura 3.16), el administrador gestiona la base de datos que contiene a todos los usuarios del sistema. En la Tabla 3.7 se muestra a detalle.




Administrado

Figura 3.16 Caso de uso: Administrar o gestionar BD Usuario.

| | |
|---|---|
| Nombre: | Gestionar o Administrar BD Usuarios |
| Descripción: | El administrador podrá agregar, modificar o eliminar, hacer consultas de de la BD usuarios |
| Actores: | Administrador |
| Precondiciones: | Estar en la BD Alumnos. |
| Flujo Normal : | 1) Administrador puede agregar / eliminar información del usuario<br>2) Administrador puede modificar la información del usuario<br>3) Administrador puede consultar información del usuario. |
| Flujo Alternativo: | 1) Si el usuario a agregar ya existe, se envía un mensaje notificando de la existencia de dicho usuario.<br>3) Si el usuario a consultar no existe se mostrara un mensaje notificando que dicho usuario no está en la BD. |

Tabla 3.7 Detalles del casos de uso para la gestión, administración de la BD Usuarios de SISAC.

**Caso de uso registro en el sistema SISAAC:** El actor que interviene en el presente caso de uso es el Usuario, el cual se registra sus datos en el sistema (ver Figura 3.17), y en la Tabla 3.8 se muestra a detalle.

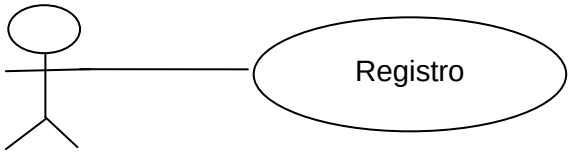



Figura 3.17 Caso de uso: Registro en sistema SISAAC.

| | |
|---|---|
| Nombre: | Registro en sistema |
| Descripción: | El usuario se registrará en el sistema para poder hacer uso del Sistema SISAAC |
| Actores: | Usuario |
| Precondiciones: | Haber hecho el pre registro |
| Flujo Normal : | 1) Ingresar nombre de usuario y contraseña<br>2) Ingresar los datos en el formulario |
| Flujo Alternativo: | 1) Si el usuario al el nombre de usuario y/o contraseña son incorrectas, mandará un mensaje de error esperando que ingresen nuevamente.<br>2) Si el usuario ingresa datos erróneos y/o deja campos en blanco es sistema enviara un mensaje de error esperando vuelva a ingresar nuevamente dichos datos. |

Tabla 3.8 Detalles del caso de uso: Registro en sistema SISAAC.

**Caso de Uso de Ingreso al sistema:** una vez que el usuario se haya registrado podrá ingresar al sistema (ver Figura 3.18), En la Tabla 3.9 se muestra a detalle.

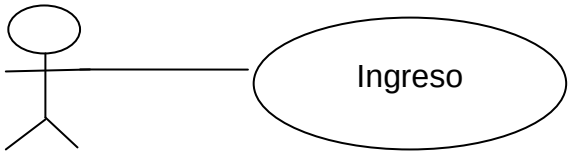



Figura 3.18 Caso de uso: Ingreso al sistema SISAAC.

| | |
|---|---|
| Nombre: | Ingreso al sistema SISAAC |
| Descripción: | El usuario ingresara al sistema |
| Actores: | Usuario |
| Precondiciones: | Haberse registrado |
| Flujo Normal : | 1) Ingresar nombre de usuario y contraseña<br>2) Ingreso a SISAAC |
| Flujo Alternativo: | 1) Si el usuario al el nombre de usuario y/o contraseña son incorrectas, mandará un mensaje de error esperando que los reingresen nuevamente. |

Tabla 3.9 Detalles del casos de uso del Ingreso al sistema SISAAC.

**Caso de uso: Crear y unirse a un grupo (Ver Figura 3.19),** el usuario una vez que ha ingresado al sistema puede crear un nuevo grupo o unirse a uno ya existente**.** En la Tabla 3.10 se muestran detalles del caso de uso ya mencionado.

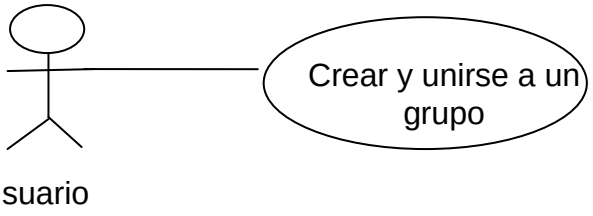



Figura 3.19 Caso de uso: Crear y Unirse a un grupo de trabajo.

| | |
|---|---|
| Nombre: | Crear y unirse a un grupo de trabajo |
| Descripción: | El usuario una vez validándose podrá unirse a un grupo ya existente o crear un nuevo grupo de trabajo. |
| Actores: | Usuario |
| Precondiciones: | Haber ingresado al sistema SISAAC |
| Flujo Normal : | Una vez que el usuario haya ingresado al sistema el usuario podrá:<br>1) Unirse a un grupo<br>2) Crear grupo<br>3) Envió o recibo mensajes dentro del grupo. |
| Flujo Alternativo: | 2) si el usuario ingresa solo ingresa dígitos o rebasa un rango mayor a 255 caracteres, el sistema enviara un mensaje notificando dicho error. |

Tabla 3.10 Detalles del casos de uso de crear y unirse a un grupo.

### 3.2.2. Diseño.

De acuerdo con el análisis de requisitos que se obtuvo a partir de los diferentes casos de uso (de la sección anterior), se realiza en las siguientes secciones el diseño que será la base sistema SISAAC. Por lo tanto, se presentan las principales Clases de en el Diagrama de Clases, los principales Diagramas de Secuencia y el Diagrama Entidad–Relación.

### 3.2.2.1. Diagramas de Clase.

El propósito del diagrama de clases es describir la estructura de la Plataforma mostrando sus clases, atributos y las relaciones que hay entre ellos. En la figura 3.20 se muestran las principales clases de la plataforma.

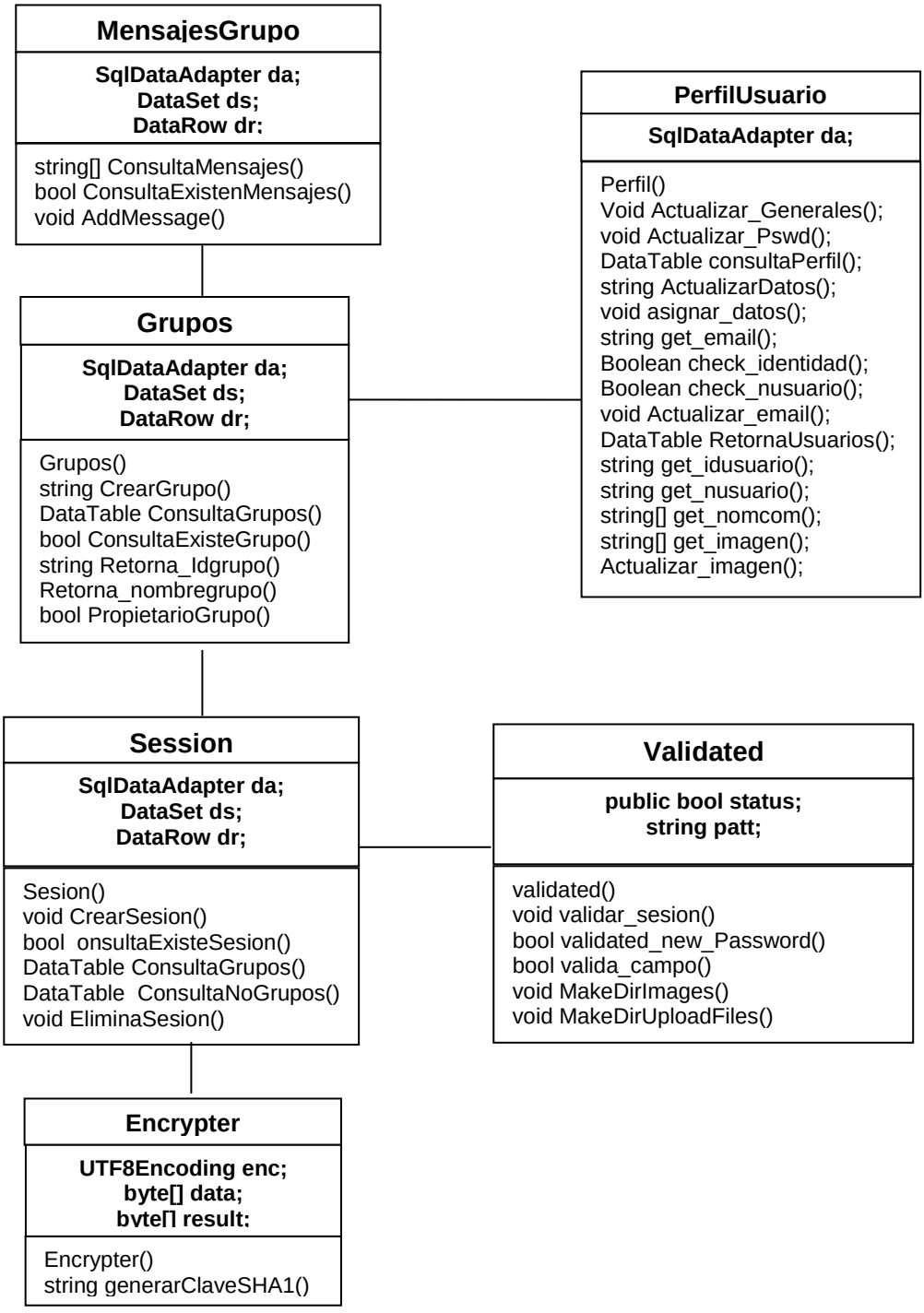



Figura 3.20 Diagrama de clases del sistema SISAAC.

### 3.2.2.2. Diagramas de Secuencia.

De acuerdo a los casos de uso realizados de la sección 3.2.1.2 del presente capítulo se generan los diagramas de secuencia que a continuación se describen.

**Diagrama de secuencia General SISAAC:** Muestra la secuencia general que presenta el Sistema SISAAC de acuerdo a los diferentes tipos de usuarios y a los diferentes recursos que intervienen en el sistema (ver Figura 3.21).

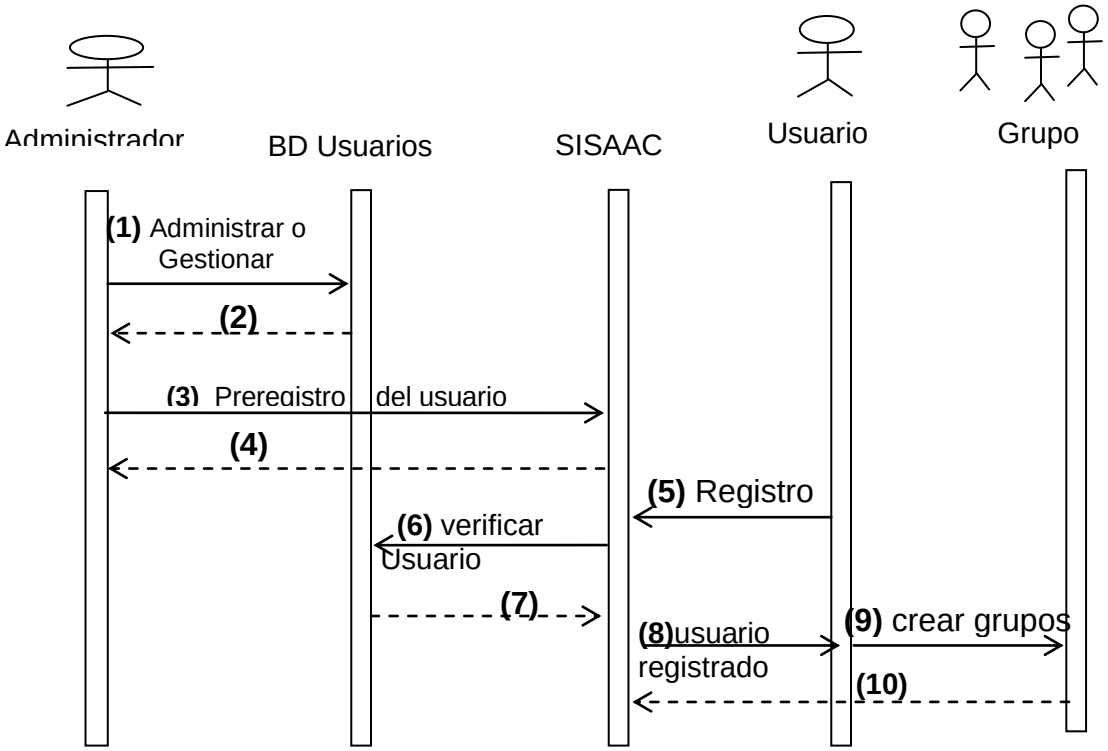



Figura 3.21 Diagrama de secuencia general del sistema SISAAC

**Diagrama de Secuencia Administrar/gestionar:** El administrador es el encargado de Administrar y gestionar la información contenida en la Base de Datos de los usuarios (ver Figura 3.22).

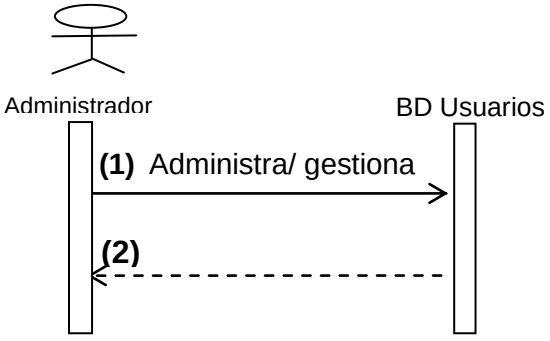



Figura 3.22 Diagrama de Secuencia: Administrar/gestionar.

**Diagrama de secuencia Preregistro**. En este caso el administrador es el encargado de hacer el pre registro de los usuarios es decir almacena la información de los usuarios en la base de datos Usuarios (ver Figura 3.23).

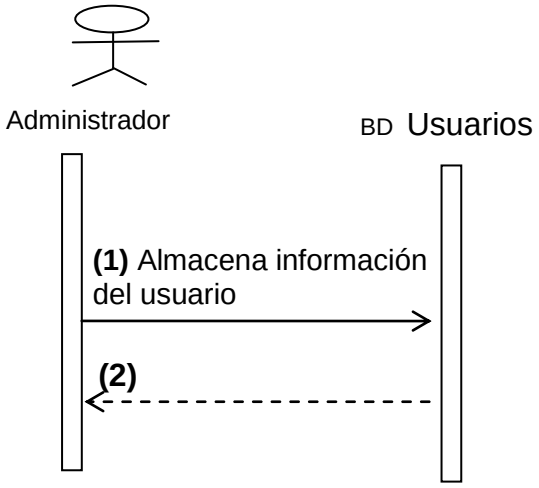



Figura 3.23 Diagrama de secuencia: Pre registro.

**Diagrama de secuencia registrar** en SISAAC el alumno deberá ingresar sus datos en el formulario en el sistema SISAAC (ver Figura 3.24).

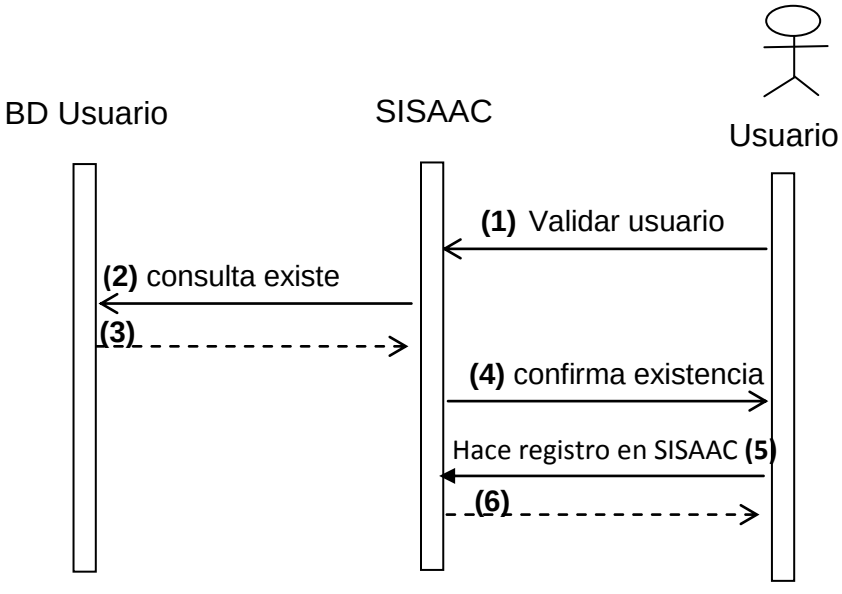



Figura 3.24 Diagrama de secuencia: registrar en SISAAC.

**Diagrama de secuencia ingreso a SISAAC**: cuando el usuario haya ingresado sus datos en el sistema podrá ingresar (ver Figura 3.25).

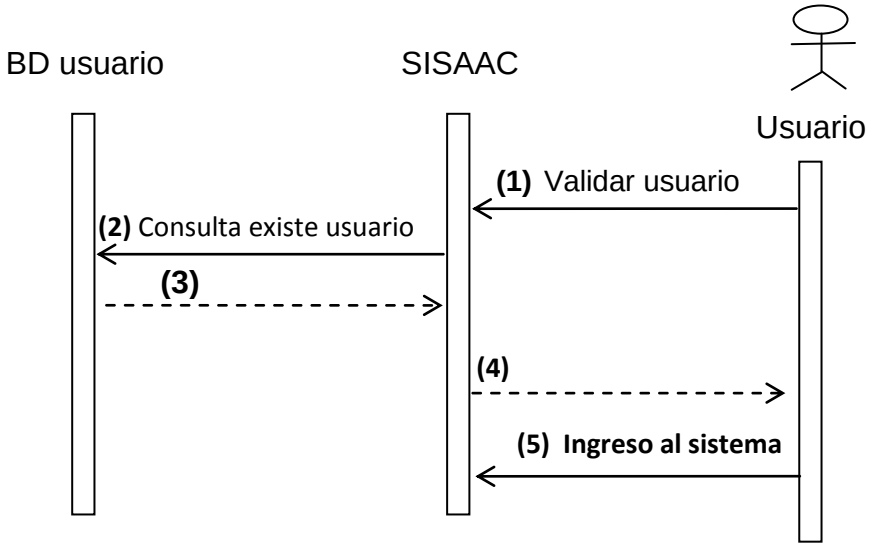



Figura 3.25 Diagrama de secuencia: ingreso a SISAAC.

**Diagrama de secuencia crear o unirse a un Grupo:** El usuario una vez ingresado al sistema podrá crear o unirse a un grupo, enviar y recibir mensajes dentro del mismo (ver Figura 3.26).

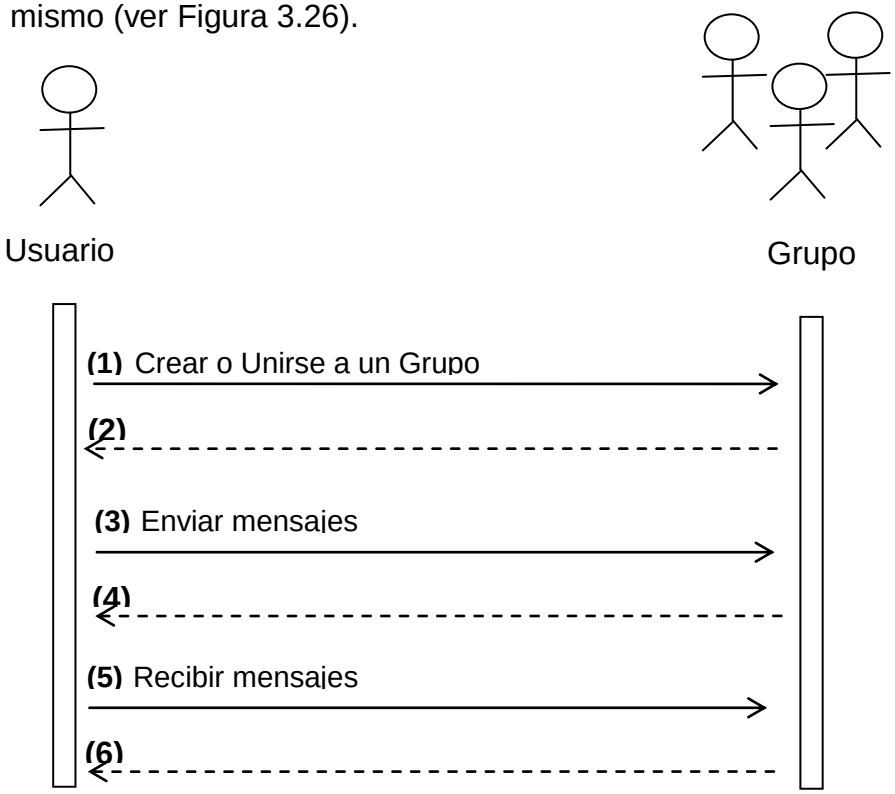



Figura 3.26 Diagrama de secuencia: crear o unirse a un Grupo.

### 3.2.2.3. Diagrama Entidad Relación.

Como teoría inicial para modelar la estructura de datos y otros conceptos que se usarán en las siguientes técnicas en el diseño de la base de datos del presente trabajo de tesis (Una Plataforma de Apoyo al Aprendizaje basada en Servicios Web). Es importante mencionar que dicho sistema requiere de herramientas las cuales se presentan en otro trabajo de tesis: “Herramientas Colaborativas basadas en Servicios Web para una Plataforma de Apoyo al Aprendizaje en la Facultad de Ciencias de la Computación”, pero se trabajo de manera conjunta en el diseño de la base de datos con las herramientas colaborativas que están montadas sobre la Plataforma, debido a que tiene una estrecha relación no fue conveniente que se trabajará de manera individual.

El Diagrama Entidad-Relación se basa en los conceptos descritos a continuación para representar el Modelo de la vida real.

Entidad: Representa una “Cosa” u Objeto.
Atributos: son propiedades que describen a cada entidad en un conjunto de entidades.
Relación: describe cierta dependencia entre entidades o permite asociación entre las mismas.

A continuación se describen las entidades y atributos.

**Entidad USUARIOS se refiere a los alumnos de la FCC que harán usó de SISAAC y contiene los siguientes atributos.**

- **IdUsuario**: identificador del usuario.
- **facultad:** se refiere a la facultad en la cual pertenecen los alumnos.
- **fechan:** fecha de Nacimiento del alumno.
- **genero:** sexo al que pertenece el alumno.
- **apmat:** apellido paterno del usuario.
- **appat:** apellido materno del usuario.
- **nombre** : nombre del usuario.
- **pwsd:** contraseña del usuario.
- **nusuario**: nombre o alias del usuario.
- **Imagen:** una imagen o foto del usuario para su perfil.
- **estado:** bandera para identificar si es la primera o más veces que ha ingresado el alumno al sistema.
- **Comentario:** algún comentario que desee hacer el alumno el cual se mostrara dentro de su perfil.
- **Email:** correo electrónico del usuario.
- **Campus:** campus al que pertenezca la facultad del alumno inscrito.
- **Carrera:** carrera en la que esté inscrito el alumno.

**Entidad GRUPO**

- **IdGrupo:** Identificador del grupo.
- **Nomgrup:** nombre del grupo.
- **Nucgrupo:** nombre del usuario quien creó el grupo.
- **descripción :** descripción del grupo a la hora de crearlo.
- **Fechacreacion:** una fecha de creación del grupo.

**Entidad MensagesGrupos:**
Idmensaje: identificador del mensaje.
IdGrupo: identificador del grupo.
Mensaje: el mensaje del alumno.
Nucmensajenombre de usuario quien creó el mensaje.
Fechacreacion: fecha de creación del mensaje.

Dadas las entidades podemos observar que existen relaciones entre usuario-grupo la cual sería Sesión porque a partir de ellas se puede obtener la información referente a los usuarios y el grupo por ejemplo: obtener los nombres de los usuarios que pertenecen algún grupo en específico, obtener información de la fecha en que el usuario creo algún grupo, etc. Enseguida de muestra la relación sesión con sus respectivos atributos:

**Relación Sesión (entre usuarios y grupo).**

IdSesion: identificador sesión.
IdUsuario: identificador del usuario.
IdGrupo: identificador del Grupo.
fecha: fecha de creación.

Además de la relación comentada anteriormente obtenemos una más entre el grupo y mensajes de grupo. En el grupo (entidad) se administra (relación) mensajes del grupo. De esta manera todas las relaciones grupo-mensagesgrupo permite obtener la información de los grupos y sus respectivos mensajes.

A partir de las entidades y sus relaciones descritas anteriormente de diseño el diagrama entidad relación (ver Figura 3.27), cabe mencionar que las entidades, atributos y relaciones de color rojo pertenece al sistema SISAAC del presente trabajo de tesis, y la parte de negro corresponde a las herramientas que pertenecen al tema de tesis “Herramientas Colaborativas basadas en Servicios Web para una Plataforma de Apoyo al Aprendizaje en la Facultad de Ciencias de la Computación”.

El esquema de la base de Datos de SISAAC se representa con un diagrama E-R el cual queda de la siguiente manera. (Ver figura 3.27).

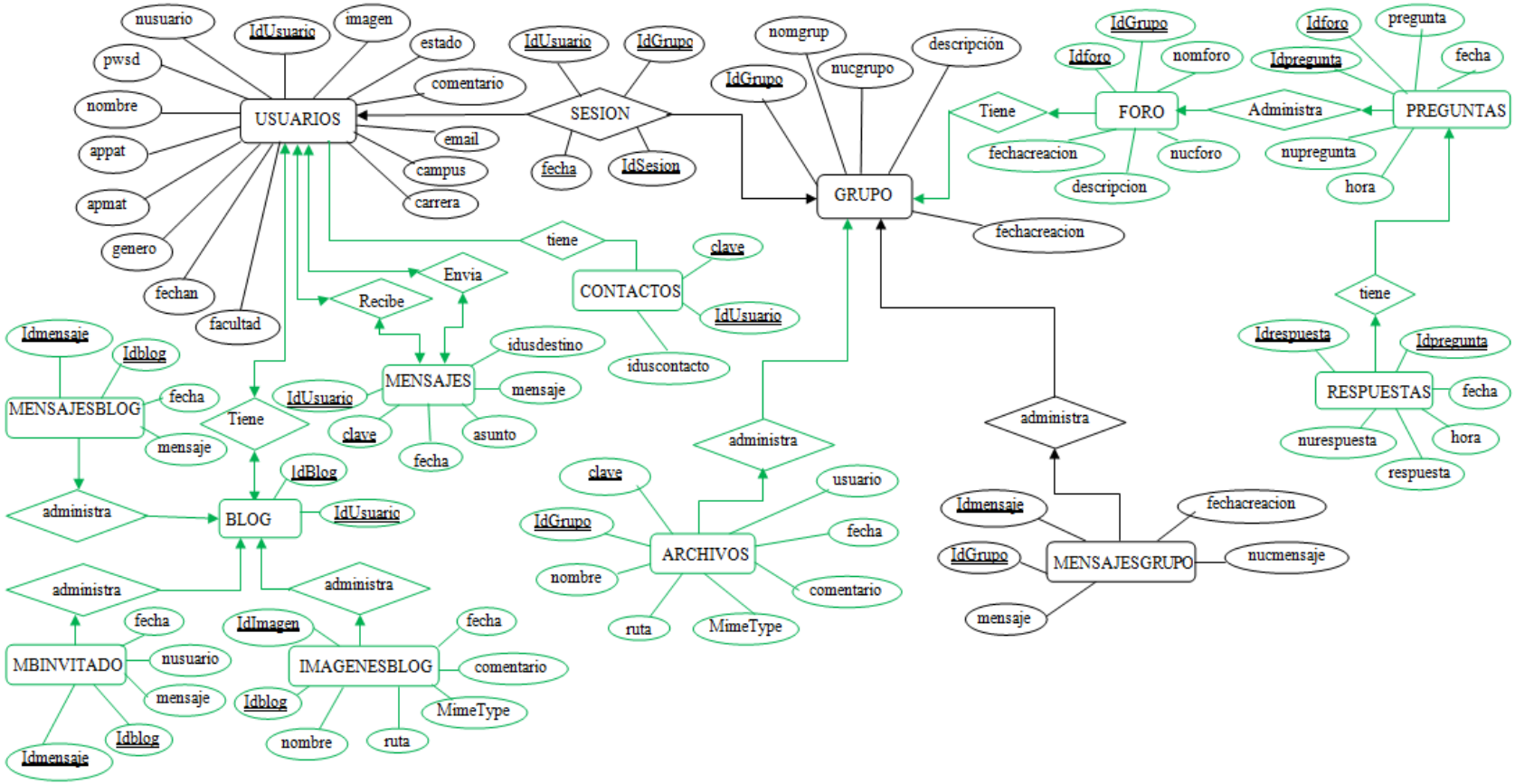



Figura 3.27: Modelo Entidad Relación de SISSAC.

El modelo relacional tiene la finalidad de dar un concepto más concreto de la forma en que debemos almacenar los datos, de ahí surge la importancia de generar un modelo relacional el cual se hace a partir de “Erwin Data Modeler en su versión 7.0” (ver Figura 3.28).

*Erwin Data Modeler* versión 7.0: es una Herramienta CASE (*Computer Aided Software Engineering*, Ingeniería de Software Asistida por Computadora), dedicada al diseño de bases de datos, la cual nos brinda la función de hacer la conversión al Modelo relacional.

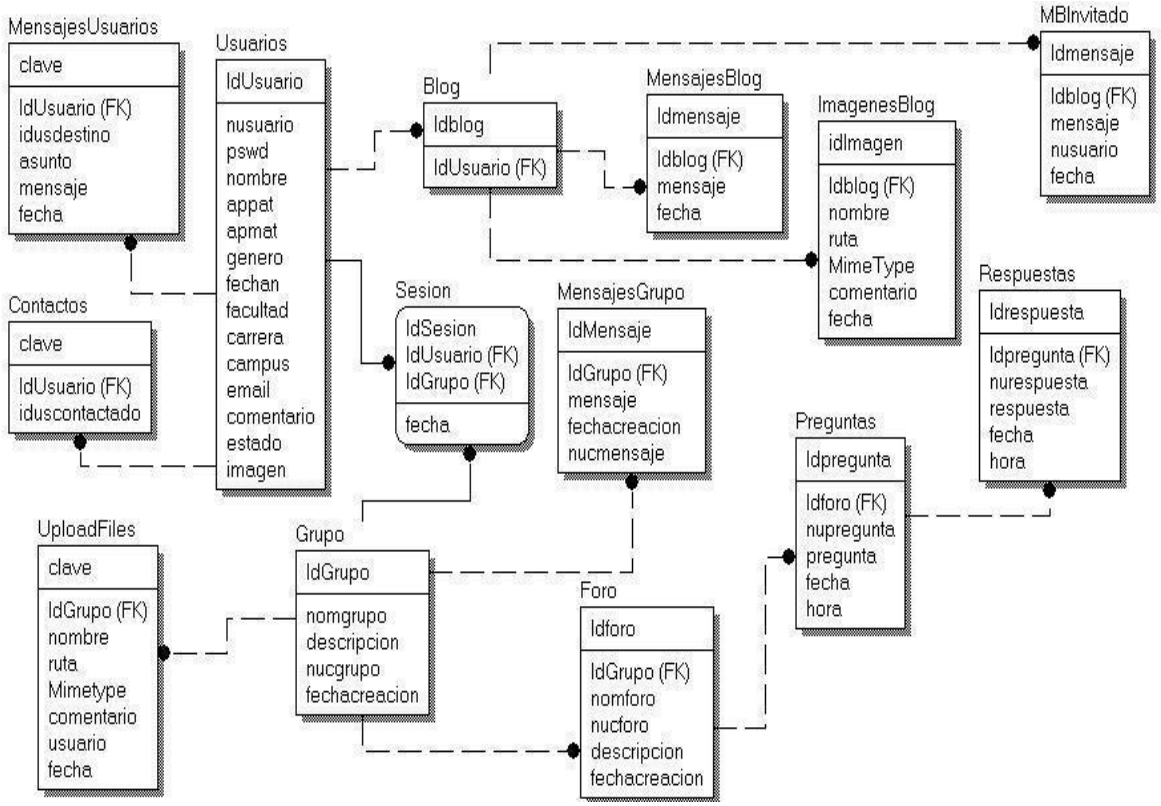



Figura 3.28. Modelo Relacional Generado por Erwin Data Modeler V7.0

### 3.2.2.4. Normalización.

La normalización es el proceso de organizar los datos en una tabla, se incluye la creación de tablas y el establecimiento de relaciones entre ellas según reglas diseñadas tanto para proteger los datos como para hacer que las bases de datos sean más flexibles al eliminar la redundancia y las dependencias.

Primera forma normal: unicidad de campo. El dominio de los atributos deben incluir solo valores atómicos (los atributos no pueden ser multivaluados ni compuestos). Cada campo de una tabla debe contener un único tipo de información. Es decir

cada campo de una tabla debe ser un dato atómico, un dato que pertenezca a un tipo de dato simple.

Segunda forma normal: clave primaria. Cada tabla debe tener un único identificador, o clave principal, que este formado por uno o más campos de la tabla. En un buen diseño de base de datos relacional, cada uno de los registros de cualquier tabla debe ser identifica

Tercera forma normal: Dependencia Funcional.

- Si Esta en segunda forma Normal.
- Eliminar los campos que no dependan de la clave principal.

Erwin Data Modeler: Garantiza incluir solo valores atómicos, que cada tabla debe tener un único identificador, además que verifica que todos los campos dependan de la clave principal, es por ello que se puede proceder al diseño de la Base de Datos.

## 4. Desarrollo del Entorno SISAAC.

En el presente trabajo de tesis se desarrolló una plataforma colaborativa llamada SISAAC (SIStema de Apoyo al Aprendizaje Colaborativo), con el objetivo de fomentar el aprendizaje en grupo donde los principales actores son los alumnos de la Facultad de Ciencias de la Computación.

SISAAC está constituido por tres partes:

1. Administrador de SISAAC: una aplicación usada por el administrador para Gestionar y administrar la base de datos que contiene la información de los usuarios.
2. La Plataforma (sistema) SISAAC: Plataforma colaborativa que permite a un usuario crear y pertenecer a grupos de trabajo, que interactúan vía web para compartir conocimiento y reforzar su aprendizaje en la FCC. Esta plataforma soporta un conjunto de herramientas colaborativas (desarrolladas en otro proyecto de tesis) para permitir la colaboración e interacción de grupo.
3. Herramientas Colaborativas: Herramientas integradas a la plataforma colaborativa SISAAC, tales como buzón de mensajes, administrador de archivos, foro para fortalecer el trabajo en grupo. Las herramientas colaborativas mencionadas anteriormente son desarrolladas en otro trabajo de tesis.

### 4.1 Administrador de SISACC.

El sistema administración de los usuarios SISAAC es una aplicación de escritorio, utilizada solo por el administrador, en la cual gestiona y administra la información contenida en la BD Usuarios de SISAAC (SIStema de Apoyo al Aprendizaje Colaborativo ).
Inicialmente para que los usuarios puedan ingresar al sistema SISAAC deberán cumplir con un pre registro realizado por el administrador, lo que quiere decir que los datos del usuario deberán ser almacenados por el administrador en la base de datos de los usuarios, el único requisito es que el usuario sea alumno de la FCC.
El sistema administración de los usuarios SISAAC se conecta a la base de datos Usuarios, una vez conectado el Administrador puede, hacer búsquedas, eliminar, actualizar o agregar un usuario o por un cierto número de usuarios contenidos en un archivo de Excel todas estas funciones la podemos ver en la interfaz del usuario (Ver Figura 4.1).

### 4.2. Desarrollo del Sistema Administrador de SISAAC.

De acuerdo al análisis de requisitos y al diseño realizado en el capitulo anterior se desarrollo el sistema administrador de SISAAC, el cual es una aplicación de escritorio, utilizando un lenguaje de programación de C#, bajo la plataforma .NET, y un gestor de Bases de Datos SQL SERVER 2008.
La interfaz del sistema y sus principales funciones se muestran en la Figura 4.1


Figura 4.1 Interfaz del sistema administrador de SISAAC.

A continuación se muestra algunos de los métodos de el archivo inicio.cs el cual ingresa a los datos con ADO.NET y C#.Net, usando una base de datos SQL Server, el resto consultar Apartado B.

**Botón conectar:** Realiza la conexión a la Base de Datos, si la conexión es exitosa realiza una consulta a la tabla Usuarios, habilita el resto de controles contenidos en la interfaz del sistema y muestra los datos empezando a ordenarlos por su ID, si no tuviera éxito la conexión manda un mensaje de error al conectarse a la Base de Datos.

```
private void btnConectar_Click(object sender, EventArgs e)
{
 string sCnn = "Server=(local); database=sisaac; integrated    security = yes";
          SqlConnection cnn = new SqlConnection(sCnn);
          string sSel = "SELECT * FROM Usuarios ORDER BY IdUsuario DESC";
          try
          {
             da = new SqlDataAdapter(sSel, sCnn);
             SqlCommandBuilder cb = new SqlCommandBuilder(da);
             cb.QuotePrefix = "[";
             cb.QuoteSuffix = "]";
             da.UpdateCommand = cb.GetUpdateCommand();
             da.InsertCommand = cb.GetInsertCommand();
             da.DeleteCommand = cb.GetDeleteCommand();
             da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
             dt = new DataTable();
             da.Fill(dt);
             foreach (Control c in this.GroupBox1.Controls)
             {  c.Enabled = true;}
             btnNuevo.Enabled = false;
             this.GroupBox1.Enabled = true;
             this.GroupBox1.Text = "Conexión realizada";
```

```
            if (dt.Rows.Count > 0)
            { btnFirst_Click(null, null); }
            else
            {
                fila = -1;
                btnActualizar.Enabled = false;
            }
        }catch (Exception ex)
        {
        MessageBox.Show("ERROR al conectar o recuperar los atos:\n" +
          ex.Message, "Conectar con la base",
         MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
```

**Botones de Movimiento:** La finalidad de los botones de movimiento es el desplazamiento de la información de la base de datos al inicio, al final de la misma o ir desplazándose de un solo usuario. Enseguida hace una llamada a la función mostrarDatos(fila) para desplegar la información.

En el botón que nos desplaza al inicio de la base de datos contiene el siguiente código.

```
private void btnFirst_Click(object sender, EventArgs e)
    {
        fila = 0;
        mostrarDatos(fila);
    }
```

El siguiente código selecciona un usuario previo al actual y muestra los datos.

```
    private void btnPrev_Click(object sender, EventArgs e)
    {
        fila = fila - 1;
        if (fila < 0) fila = 0;
        mostrarDatos(fila);
    }
```

El código que mostramos a continuación hace posible el desplazamiento del usuario siguiente y muestra sus datos.

```
     private void btnNext_Click(object sender, EventArgs e)
    {
        int uf = dt.Rows.Count - 1;
        fila = fila + 1;
        if (fila > uf) fila = uf;
        mostrarDatos(fila);
    }
```

En siguiente código se encuentra contenido en el botón que desplaza al final de la base de datos y muestra la información del usuario.

```
    private void btnLast_Click(object sender, EventArgs e)
    {
        fila = dt.Rows.Count - 1;
        mostrarDatos(fila);
    }
```

Función encargada de mostrar los datos en los textbox,

```
private void mostrarDatos(int f)
    {
        int uf = dt.Rows.Count - 1;
        if (f < 0 || uf < 0) return;
        DataRow dr = dt.Rows[f];
        txtIdUsuario.Text = dr["idusuario"].ToString();
        txtpswd.Text = dr["pswd"].ToString();
        txtNombre.Text = dr["Nombre"].ToString();
        txtappat.Text = dr["appat"].ToString();
        txtapmat.Text = dr["apmat"].ToString();
        btnActualizar.Enabled = true;
        btnEliminar.Enabled = true;
    }
```

**Actualizar.** Permite actualizar la base de datos despues de haber hecho modificaciones a la información

```
private void btnActualizar_Click(object sender, EventArgs e)
    {
        if (fila < 0 || fila > dt.Rows.Count - 1) return;

        DataRow dr = dt.Rows[fila];
        asignarDatos(dr);
        try
        {
            da.Update(dt);
            dt.AcceptChanges();
            MessageBox.Show("Datos del Usuario Actualizados ...!!!");
        }
        catch (DBConcurrencyException ex)
        {
            MessageBox.Show("Error de concurrencia:\n" + ex.Message);
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
```

**Agregar :** Permite agragar la información en la base de datos capturados en la intefaz del sistema, el código se muestr a enseguida.

```
private void btnNuevo_Click(object sender, EventArgs e)
    {
        // Crear un nuevo registro
        DataRow dr = dt.NewRow();
        // Asignar los datos de los textbox a la fila
        asignarDatos(dr);

        // Añadir la nueva fila a la tabla
        dt.Rows.Add(dr);
        // Guardar físicamente los datos en la base
        try
        {
            da.Update(dt);
            dt.AcceptChanges();
            // Si es el primer registro de la base,
            // volver a leer los datos para actualizar los IDs
            if (Convert.ToInt32("0" + dr["idusuario"].ToString()) == 0)
            {
                dt = new DataTable();
                da.Fill(dt);
            }
```

```
            // Posicionarlo en la última fila
            btnLast_Click(null, null);
        }
        catch (DBConcurrencyException ex)
        {
            MessageBox.Show("Error de concurrencia:\n" + ex.Message);
        }
        catch (Exception ex)
        { MessageBox.Show(ex.Message);}
        btnNuevo.Enabled = false;
        btnlimpiar.Enabled = true;
        btnActualizar.Enabled = true;
        btnEliminar.Enabled = true;
        btnExcel.Enabled = true;
        MessageBox.Show("Usuario Dado de Alta en el Sistema ...!!!");
    }
```

**Los check books y los text box.** El fin de los chekBox son habilitar o seleccionar los textBox para que se pueda almacenar la información contenida en la base de datos de los usuarios.

```
private void asignarDatos(DataRow dr)
    {
        if(checkBox1.Checked)
            dr["idusuario"] = txtIdUsuario.Text;
        dr["nusuario"] = txtNombre.Text;
        if (checkBox2.Checked)
            dr["pswd"] = generarClaveSHA1(txtpswd.Text);
        if (checkBox3.Checked)
            dr["nombre"] = txtNombre.Text;
        if (checkBox4.Checked)
            dr["appat"] = txtappat.Text;
        if (checkBox5.Checked)
            dr["apmat"] = txtapmat.Text;
        if (!checkBox6.Checked)
            dr["estado"] = "false";
        else
            dr["estado"] = "truee";
    }
```

Podemos observar en la línea del checkBox2 se invoca la función generarClaveSHA1(txtpswd.Text) se pasa como parámetros la información contenida en txtpswd.Text que corresponde a la contraseña dicha función es la encargada de encriptar las contraseñas de los usuarios dejando una cadena de 40 caracteres .

```
public string generarClaveSHA1(string pass)
    {
        UTF8Encoding enc = new UTF8Encoding();
        byte[] data = enc.GetBytes(pass + "sisaac-fcc-buap");
        byte[] result;
        SHA1CryptoServiceProvider sha = new SHA1CryptoServiceProvider();
        result = sha.ComputeHash(data);
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < result.Length; i++)
        {
```

```
            if (result[i] < 16)
            {
                sb.Append("0");
            }
            sb.Append(result[i].ToString("x"));
        }
        return sb.ToString().ToUpper();
    }
```

En la línea donde se encuentra el checkBox 6 podemos observar estado , no es más que una bandera que ayuda a checar si es la primera vez que el usuario ha ingresado al sistema.
cuando se encuantra en false significa que el usuario no ha ingresado al sistema anteriormente y deberá llenar el formulario del registro, por el contrario truee significa que el usuario ya ha ingresado anteriormente y lo redirecciona a la página de inicio del sistema.SISAAC.

**Eliminar.** Elimina un usuario de la base de datos que se ha seleccionado en la intefaz .

```
private void btnEliminar_Click(object sender, EventArgs e)
    {
        // Eliminar la fila actual
        if (fila < 0 || fila > dt.Rows.Count - 1) return;

        try
        {
            // Eliminar la fila de la tabla
            dt.Rows[fila].Delete();
            // Actualizar físicamente la base de datos
            da.Update(dt);
            // Aceptar los cambios en la copia local
            dt.AcceptChanges();
            txtIdUsuario.Text = txtpswd.Text = txtNombre.Text = txtappat.Text = txtapmat.Text = "";
            MessageBox.Show("Usuario Eliminado del Sistema ...!!!");
        }
        catch (DBConcurrencyException ex)
        {   MessageBox.Show("Error de concurrencia:\n" + ex.Message); }
        catch (Exception ex)
        {   MessageBox.Show(ex.Message);  }
    }
```

Para la importación de datos se hizo uso de ADO.NET para tener acceso a los datos, así como diferentes librerías y controladores brindados por la plataforma ASP.NET para tener un correcto funcionamiento de importación de datos.

La presente aplicación está configurada para trabajar a 32 bits (x86), además hace uso de la librería microsoft excel 12.0 esta librería hace posible la conexión entre sqlserver y excel versión 2007, es importante mencionar que para cada versión de excel se utiliza deferente librería.

## 4.3. Desarrollo de SISAAC.

Para el desarrollo de SISAAC (SIStema de Apoyo al Aprendizaje Colaborativo) se utiliza tres hojas de estilo en cascada, para darle una mejor estructura y apariencia a los documentos (páginas web), la primera de ellas para la validación del usuario antes de ingresar a SISAAC, posteriormente otra para el registro del usuario y por último la pagina principal.

Así como también en el desarrollo de SISAAC se implementaron diversos servicios web, como ya se explicó un servicio web es un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Los servicios web son accesibles desde HTTP y siguen una serie de estándares cuyos responsables son las organizaciones OASIS y W3C.

Para fines prácticos, los servicios web involucrados en SISAAC se desarrollaron en ASP.NET y C#.NET, por la gama de herramientas que el framework .NET provee al programador y SQLSERVER 2008 (ADO.NET) para la base de datos.

ASP.NET se encarga de los diferentes procesos para la descripción (WSDL), la comunicación (SOAP) y Repositorio de los Servicios Web (UDDI), posteriormente se explica un Servicio Web utilizado en SISAAC y el resto de Servicios Web se puede ver en el Apéndice B.

### 4.3.1. Interfaz para validar datos del Usuario.

La interfaz para la validación del usuario (ver Figura 4.2) contiene dos TextBox, el TextBox superior es asignado para Id del usuario, y el que se encuentra en la parte inferior se asigna para el nombre del usuario.
La validación consta del Id de usuario el cual corresponde a la matrícula que asigna la universidad al ingresar a la misma, la contraseña es tomada de la matrícula y corresponde a los 6 últimos dígitos de ella y otra plantilla para realizar el registro.
La parte del código que hace la validación del usuario corresponde a un Servicio Web que a través de los métodos SearchUser(checa si existe el usuario en el sistema ) y Estado_Usuario(verifica el estado del usuario, si es *true* abre la interfaz de inicio del sistema en caso contrario se procede al registro de usuarios) hace posible el ingreso al sistema, ambos metodos corresponden al servicio web AccessWS.asmx.

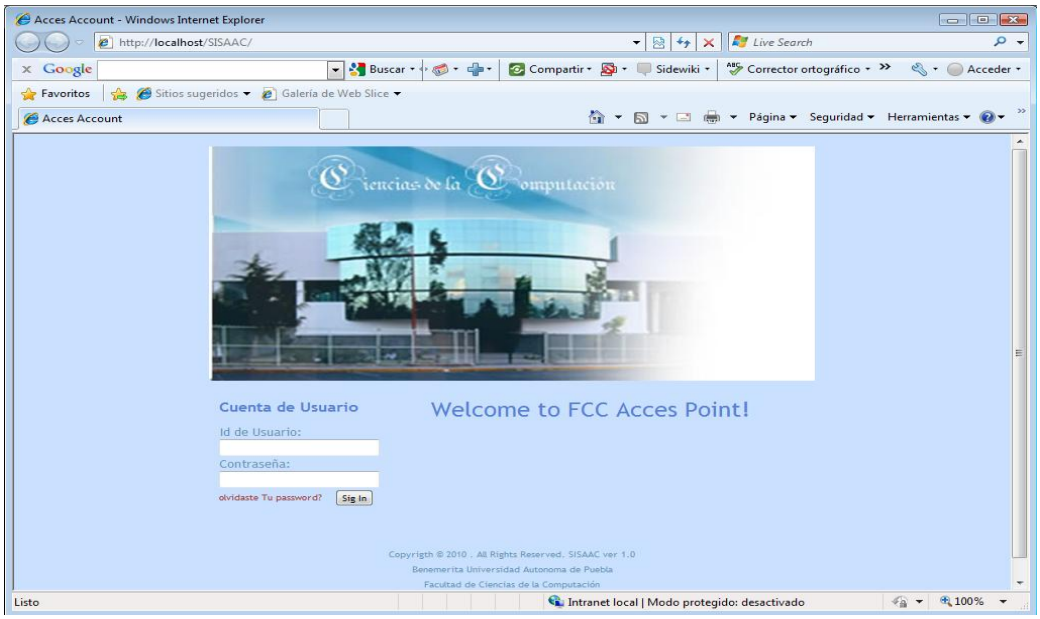



Figura 4.2 Interfaz para Validar usuario

### 4.3.2. Interfaz de para el registro del usuario.

En la Figura 4.3 se muestra la interfaz correspondiente al llenado del formulario con los datos del Alumno, cada campo esta previamente validado para tener una mejor organización y control de la información evitando los posibles errores que el usuario pueda tener al momento de escribir sus datos.

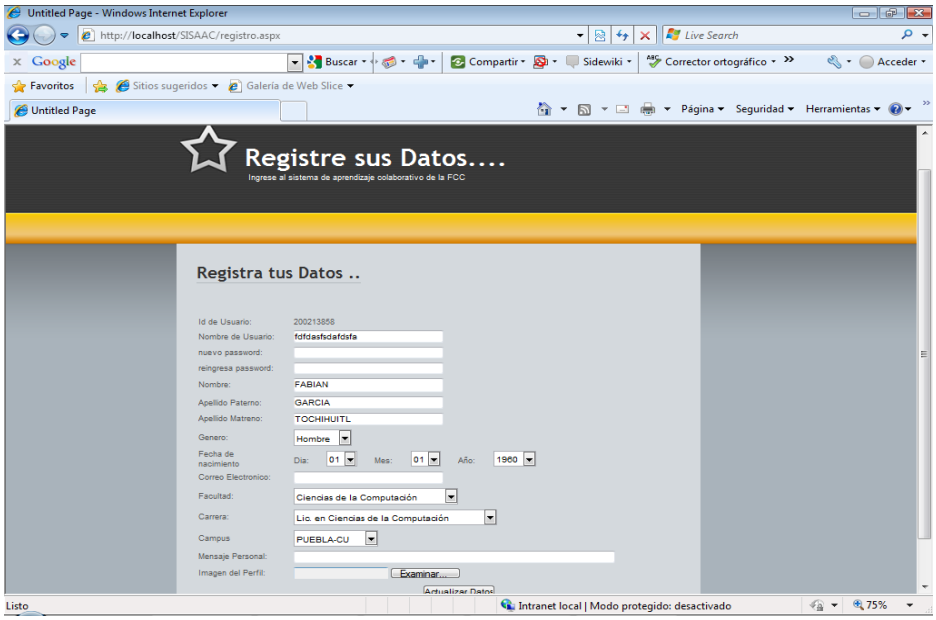



Figura 4.3 Interfaz para Formulario de Registro

### 4.3.3 Interfaz de usuario centrado en grupos.

Para la interfaz de usuario centrado en grupos de SISAAC (ver Figura 4.4), se utilizó CSS con el cual definimos los fondos de imágenes, formato de las fuentes y colores de fondo, además se trabajo con JavaScript en conjunto con DOM; el primero para llamadas asíncronas ya que se ejecuta del lado del cliente, dando una mejor funcionalidad al sistema. Con los objetos DOM se modela el documento o página web, y algunos elementos que pueda contener la propia página, como párrafos, divisiones, tablas y formularios, etc. Los elementos de DOM se pueden acceder, por medio de JavaScript, a cualquiera de estos elementos, es decir a sus correspondientes objetos para alterar sus propiedades o invocar a sus métodos.

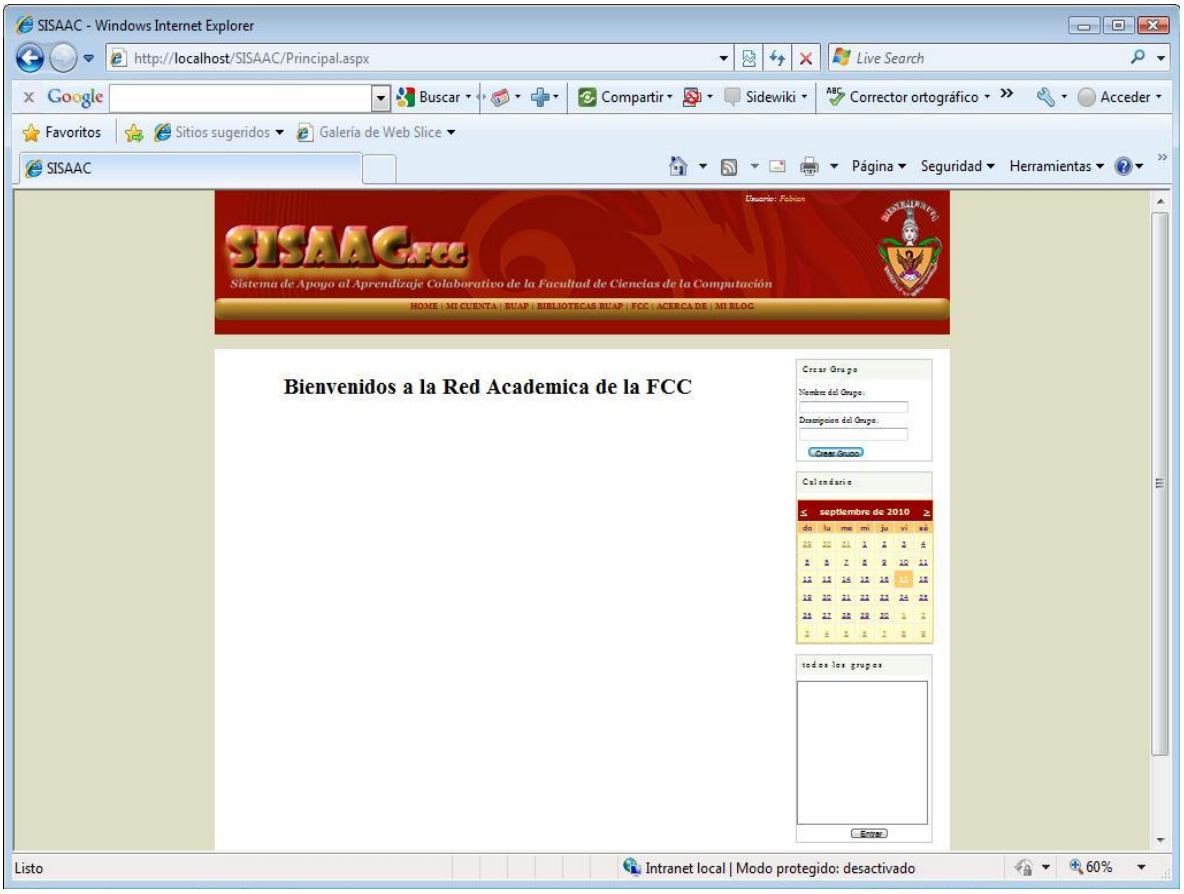



Figura 4.4 Interfaz para de SISAAC centrado en Grupos.

A continuación mostramos un ejemplo de CSS con un fragmento de la hoja de estilo *perfil.css,* en donde definimos las dimensiones de la página y sus elementos como secciones, también define el fondo de la imagen, colores, tipo de fuente, y colores de fondo de página principal de SISAAC (ver Figura 4.4). El resto de la plantilla puede consultarlo en el Apéndice B.

Comenzamos ajustando a cero el margen y el relleno del elemento *body*(cuerpo) para lograr la compatibilidad con la configuración predeterminada de los diversos

navegadores, cada valor en hexadecimal en CSS pertenece a un color. En específico, además definimos el color de fondo de la página con un valor en hexadecimal.

```
body {
   margin-top: 0px;
        padding: 0;
   background:#dfdfc5;
   color:#000000;
}
```

Se hace un ajuste de las dimensiones que ucupa el encabezado asi como el valor del margen y el color de fondo de dicho encabezado.

```
#encabezado {
        width: 1075px;
        height:200px;
        margin: 0 auto;
        background:#960f00;
}
```

En el selector de Id contenedor definimos las dimensiones, ancho y alto, el valor del margen , el color de fuente y de fondo de los elementos, de lo que contendrá el contenedor principal en cual contiene más elementos dentro de éste.

```
/*contenedor principal*/
#contenedor
{
   clear:both;
        width: 1075px;
        height:auto;
        margin: 0 auto;
        color:#000000;
        background-color:#ffffff;
}
```

En el siguiente fragmento (selector de Id logo) corresponde al logotipo SISAAC, de igualmanera que en los selectores anteriores definimos las dimensiones, pero insertamos una imágen de fondo con su respectiva url (dirección donde se encuentra alojada dicha imágen).

```
#logo {
   width:1075;
   height:150px;
   background:url(../images/logo_sisaac.png);
}
/*Barra de menus y usuario*/
#cabecera {
        height: 28px;
        margin: 0 auto;
        background: #960f00;
}
```

Al igual que los anteriores selectores de ID, se definen los atributos que contentiene cada selector como las dimensiones de las imágenes que se insertan, alineación de texto el tipo de fuente etc., el siguiente selector de Id define las los atributos que contiene el menu de SISAAC.

```
/*Barra de menus*/
#menu{
```

```
clear:both;
    width:1075px;
    height:25px;
    background:url(../images/barra_menus.png);
    text-align:center;
    padding-top:5px;
    font: 13px Georgia, "Times New Roman", Times, serif;
    text-transform: uppercase;
}
```

En el selector cuerpo que corresponde al área de trabajo de la página principal, en la cual se mostrarán los diversos mensajes que el usuario creará. A continuación se muestra como define el tamaño, el margen, el relleno, el color de fuente etc.

```
/*Area de trabajo*/
#cuerpo{
width:800px;
margin-left: 0px;
padding: 12px 0px 10px 0px;
background-color :#ffffff;
float:left;
}
```

En el selector lateral corresponde a la definicion de dimenciones, margen color, etc., de barra lateral derecha de la página principal donde se permite crear grupos, muestra un calendario y lista todos los grupo dados de alta en SISAC (ver Figura 4.4) se comienza definiendo color de fondo, un elemento flotante a la deracha, las dimenciones, tamaño del margen y un tamaño de fuente.

```
#lateral{
background-color: #ffffff;
float:right;
width:200px;
height:auto;
margin-right:25px;
font-size:13px;
}
```

Los siguentes selectores de clase estan contenidos dentro del selector ID lateral y corresponde al encabezado de cada elemento de la barra lateral que son la parte de crear grupos, calendario, y listado de grupos es decir define las propiedades que contiene el titulo de cada elemento de la barra lateral derecha (ver Figura 4.5)

```
.titlat{
background-color:#f4f8f0
    color:#000000;
    font-size:13px;
    padding: 7px 3px 7px 8px;
    font-weight : normal;
    letter-spacing : 2px;
    margin: 0px 0px 9px 0px;
    font-family:Times New Roman;
}
.cuerpolateral{
padding: 5px 4px 13px 10px;
background-color: #ffffff;
}
```

En el código anterior explicamos un fragmento de la hoja de estilos en cascada de la página principal ahora haremos una explicación de lo que contiene la interfaz

con respecto a la manera de crear grupos y unirse a un grupo ya existente, así como de los métodos que se invocan del servicio web. Correspondiente a crear Grupos.

Para que el usuario pueda crear un nuevo grupo en SISAAC debe ingresar el nombre del grupo a crear y una descripción del tema que se abordará en dicho grupo, esta opción se encuentra en la parte lateral derecha y dice “Crear Grupos” (ver Figura 4.4). El servicio web que se utiliza es Grupos.asmx, que en primer lugar realiza una consulta para verificar si existe algún otro grupo con el mismo nombre mediante el método ConsultaExisteGrupo, si resultara positivo envia un mensaje de error impidiendo su creación, en caso contrario se crea (utilizando el método CrearGrupo) redireccionando al usuario al nuevo grupo creado.

Para integrarse a un grupo ya existente, se selecciona el grupo de un ListBox, que se encuentra en la parte lateral derecha de la página principal (ver Figura 4.4 parte inferior), que contiene todos los grupos existentes. Esto se hace mediante el servicio web Grupos con su método ConsultaGrupos del servicio web anteriormente mencionado. Enseguida seleccionamos el grupo al que deseamos unirnos a él, por último un clic al Boton entrar e inmediatamente nos redirecciona al grupo deseado. En la Figura 4.4 se encuentra un calendario en el centro de la parte lateral derecho.

En la barra superior corresponde a la barra de navegación, que contiene vínculos hacia sitios web de la Universidad, como la página oficial de la Universidad (BUAP), a la página oficial de la FCC, etc., así como para interactuar dentro de SISAAC. El siguiente código en un script en JavaScript que pertenece a la barra de navegación.

```
function menu_nav(){
   var alias = document.getElementById("lbl_nusuario").innerHTML;
   var id = document.getElementById("lbl_idusuario").innerHTML;
   var ligas = '<a href="Principal.aspx?id='+id+'&alias='+alias+'" class="enlacenav"><b> Home </b></a> | ';
   ligas += '<a href="usercp.aspx?id=' + id + '&alias=' + alias + '" class="enlacenav"><b> MI CUENTA</b></a>
| ';
            ligas += '<a href="http://www.buap.mx" target="_new" class="enlacenav"><b> BUAP </b></a> | ';
            ligas += '<a href="http://www.bibliotecas.buap.mx" target="_new" class="enlacenav"><b> BIbliotecas
BUAP </b></a> | ';
            ligas += '<a href="http://www.cs.buap.mx" target="_new" class="enlacenav"><b> FCC </b></a> | ';
            ligas += '<a href="success.aspx?id=' + id + '&alias=' + alias + '" class="enlacenav"><b> Acerca de
</b></a> | ';
            ligas += '<a href="Blog.aspx?name=' + alias + '&id_ses=' + id + '&alias_ses=' + alias + '"
class="enlacenav"><b> Mi Blog </b></a> | ';
            ligas += '<a href="/sisaac" class="enlacenav"><b> Cerrar Sesion </b></a> ';
            document.getElementById("menu").innerHTML = ligas;
}
```

### 4.3.3.1. Descripción de Servicios Web utilizados por SISAAC.

En la presente sección hacemos una breve explicación de los elementos que contiene el Servicio Web Grupos.asmx, tomando solo un método (*ConsultaGrupos*) para fines prácticos, si desea consultar los métodos web restantes de cada servicio consulte el apéndice B.

Cuando definimos un Servicio Web realmente estamos definiendo una clase y dentro de esa clase indicamos que métodos (funciones) queremos exponer como parte del servicio web tendremos que usar atributos que están definidos en el espacio de nombres System.Web.Services.

```
using System.Web.Services;
using System.Configuration;
```

los siguentes espacios de nombres son para poder ingresar a una base de datos SQLSERVER en las que están definidas las clases que utilizamos durante el desarrollo.

```
using System.Data.SqlClient;
using System.Data;

using System;
```

enseguida se definimos la clase a la que vamos a aplicara el atributo WebServiceAttribute para indicar el espacio de nombres del servicio web y una descripción de lo que hace este servicio web deben estar derivadas de System.Web.Services.WebService.

```
[WebService(Namespace = "http://tempuri.org/")]
```

Al crear un servicio web, tenemos que usar el atributo WebServiceBinding el cual permite especificar una o más uniones de forma que se puedan crear componentes conforme al *WS-I Basic Profile.* WS-I Basic Profile 1.1 es una especificación que consta de un conjunto de especificaciones de servicios web no propietario junto con aclaraciones, ajustes, interpretaciones y ampliaciones de las especificaciones que promueven la interoperabilidad, como SOAP y WSDL.

```
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
```

Descomentar la siguiente línea de código para indicar que el servicio web será invocado por JavaScript o ASP.NET AJAX.

```
// [System.Web.Script.Services.ScriptService]

public class Grupos : System.Web.Services.WebService
{
```

Para poder ingresar a la tabla de la base de datos usaremos un DataAdapter de SQL

```
    protected SqlDataAdapter da;

    protected DataSet ds;
    private DataRow dr;
    protected SqlConnection myConnection = new SqlConnection();

    public Grupos () {  }
```

Ahora definimos los métodos que se expondrá en el servicio web, a estos métodos se les llama métodos web y se aplica el atributo WebMethodAttribute, al que añadimos una descripción de dicho método para saber que es lo que hace, por ejemplo: el método Web ConsultaExisteGrupo recibe como parámetros una cadena y devuelve un valor boleano *True* o *False* según sea el caso. En cada servicio se pueden definir uno o mas métodos web cada uno con sus propias operaciones.

```
[WebMethod(Description = " Metodo que retorna un datatable con todos los   grupos")]
    public DataTable ConsultaGrupos()
    {
        string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
        SqlConnection myConnection = new SqlConnection(strConnectionString);

        string strCommandText = "SELECT idgrupo,nomgrupo FROM Grupo ";
        da = new SqlDataAdapter(strCommandText, myConnection);
        SqlCommandBuilder cb = new SqlCommandBuilder(da);
        da.UpdateCommand = cb.GetUpdateCommand();
        da.InsertCommand = cb.GetInsertCommand();
        da.DeleteCommand = cb.GetDeleteCommand();
        da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
```

para conservar los datos, vamos a usar un DataTable, por ello creamos un nuevo objeto de ese tipo para después usarlo con el DataAdapter.

```
      DataTable dt= new DataTable();
```

Enseguida llenamos el DataTable con los datos solicitados. Para ello utilizamos el método Fill del DataAdapter. Es recomendable utilizar try/catch para interceptar los posibles errores que puedan producir.

```
        try
        {
            da.Fill(dt);
        } catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
            return null;
        }
        finally
        {
            if (myConnection != null)
                myConnection.Close();
        }
```

Por último devolvemos el DataTable con los datos.

```
        return dt;
    }
```

Como cada servicio web tiene prácticamente los mismos elementos descritos anteriormente se procederá a explicar de manera general los resultados esperados de acuerdo al servicio y método que se exponen.

El servicio web AccessWS.asmx corresponde a la validación del usuario comenzamos definiendo los espacios de nombres de los cuales vamos a hacer uso de sus atributos definidos en ellos. El método del servicio seb para validación del usuario presentado a continuación tiene la finalidad de retornar un valor booleano (true o false) para indicar si IdUsuario(IdUsuario) y la contraseña(pswd) corresponden a los datos ingresados por el usuario, si resulta verdadero retorna true en caso contrario retorna false. La interfaz encargada de invocar al servicio web antes mencionado se muestra en la Figura 4.5.


Figura 4.5: usuario validando Id de Usuario y su contraseña para ingresar a SISAAC.

```
using System.Web.Services;
using System.Configuration;
using System.Data.SqlClient;
using System.Data;
using System;
[WebService(Namespace = "http://microsoft.com/webservices/", Description = "Acceso a la base de datos Alumos (local) desde un servicio Web")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
// [System.Web.Script.Services.ScriptService]
public class AccessWS : System.Web.Services.WebService
{
    private SqlDataAdapter da;
    private DataSet ds;
    private SqlConnection myConnection = new SqlConnection();
    [WebMethod(Description = " Metodo que retorna un mensaje de texto indicando si se encontrado el registro buscado")]

    public bool SearchUser(string id,string pass){
        string strConnectionString =
```

```
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
        SqlConnection myConnection = new SqlConnection(strConnectionString);
    string  strCommandText  =  "SELECT  IdUsuario,pswd  FROM  Usuarios  WHERE  IdUsuario  =
'"+id+"' AND pswd = '"+pass+"'";

        da = new SqlDataAdapter(strCommandText, myConnection);
        ds = new DataSet();
        try
        {
            da.Fill(ds, "Usuarios"); ;
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);

        }
        finally
        {
            if (myConnection != null)
                myConnection.Close();
        }
        if (ds.Tables["Usuarios"].Rows.Count > 0)
            return true;
        else
            return false;
    }
```

Enseguida mostramos el código que corresponde al servicio web PerfilUsuarios.asmx, el cual podemos observar que contiene el método consultaPerfil, el cua recibe como parámetros un ID que corresponde al ID de usuario, hace una consulta con el parámetro antes mencionado a la base de datos y retorna un DataTable con la información contenida del usuario, y manipulada en la interfaz para mostrarlos en sus correspondientes etiquetas (Label) (ver Figura 4.6).

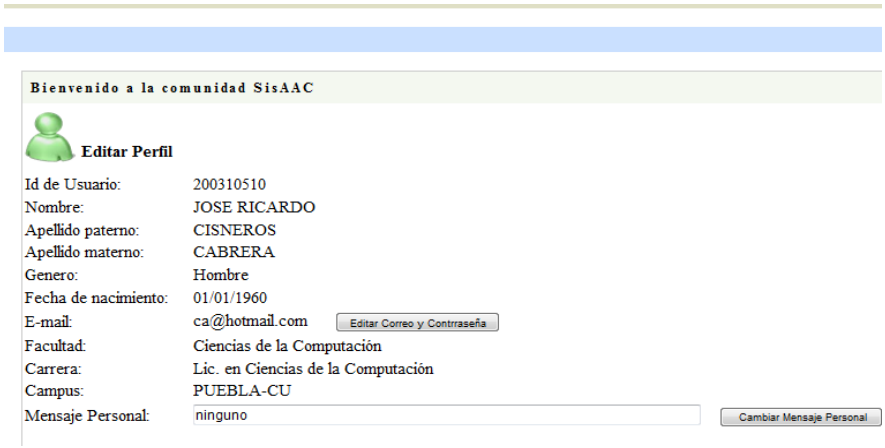



Figura 4.6 Muestra los datos del perfil de usuario.

```
using System.Web.Services;
using System.Configuration;
```

```
using System.Data.SqlClient;
using System.Data;
using System;
namespace Samples.Aspnet
{
   [WebService(Namespace = "http://tempuri.org/")]
   [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
   [System.Web.Script.Services.ScriptService]
   public class Perfil : System.Web.Services.WebService
   {
      protected SqlDataAdapter da;
      protected DataSet ds;
      protected SqlConnection myConnection = new SqlConnection();
     public Perfil()
      {       }
      [WebMethod(Description = " Metodo que retorna un datatable con el perfil de un Usuario")]
      public DataTable consultaPerfil(string id)
      {
         string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
         SqlConnection myConnection = new SqlConnection(strConnectionString);
         string strCommandText = "SELECT * FROM Usuarios " + " WHERE IdUsuario = '" + id + "'";
         da = new SqlDataAdapter(strCommandText, myConnection);
         SqlCommandBuilder cb = new SqlCommandBuilder(da);
         da.UpdateCommand = cb.GetUpdateCommand();
         da.InsertCommand = cb.GetInsertCommand();
         da.DeleteCommand = cb.GetDeleteCommand();
         da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
         DataTable dt = new DataTable();
         try
         {  da.Fill(dt);}
           catch (Exception ex){
            Console.WriteLine(ex.Message);
            return null;
         }
         finally
         {
            if (myConnection != null)
               myConnection.Close();
         }
         return dt;
      }
```

A continuación mostramos el Código correspondiente al servicio web Sesion.asmx, dentro de dicho servicio web encontramos el método crear sesión el cual tiene la finalidad de crear una sesión para el usuario, como primer caso recibe tres cadenas como parámetros que son: IdUsuario, IdGrupo,fecha, hace una consulta a la base de datos, crea un objeto DataTable, enseguida llenamos el DataTable con los datos solicitados, hacemos la asignacion de los datos que nos pasaron como parámetros y los asignamos dentro de los registros correspondientes a cada valor (ver Figura 4.7).


Figura 4.7 Muestra el usuario Cisneros Cabrera esta en una sesión.

```
sing System.Web.Services;
using System.Configuration;
using System.Data.SqlClient;
using System.Data;
using System;

namespace Samples.Aspnet
{
   [WebService(Namespace = "http://tempuri.org/")]
   [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
   [System.Web.Script.Services.ScriptService]
   public class Sesion : System.Web.Services.WebService
   {
      DataRow dr;
      SqlDataAdapter da;
      DataSet ds;
      protected SqlConnection myConnection = new SqlConnection();

      public Sesion() { }
      [WebMethod(Description = " Metodo que Crea una sesion de Usuario en un Grupo")]
      public void CrearSesion(string IdUsuario, string IdGrupo, string fecha)
      {
         string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
         SqlConnection myConnection = new SqlConnection(strConnectionString);
         string strCommandText = "SELECT * FROM Sesion";
         da = new SqlDataAdapter(strCommandText, myConnection);
         SqlCommandBuilder cb = new SqlCommandBuilder(da);
         cb.QuotePrefix = "[";
         cb.QuoteSuffix = "]";
         da.UpdateCommand = cb.GetUpdateCommand();
         da.InsertCommand = cb.GetInsertCommand();
         da.DeleteCommand = cb.GetDeleteCommand();
         da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
         DataTable dt = new DataTable();
         try
         {
            da.Fill(dt);
            dr = dt.NewRow();
            dr["IdUsuario"] = IdUsuario;
            dr["IdGrupo"] = IdGrupo;
            dr["fecha"] = fecha;
            dt.Rows.Add(dr);
            da.Update(dt);
            dt.AcceptChanges();
         }
         catch (Exception ex)
         { Console.WriteLine(ex.Message);}
         finally{
            if (myConnection != null)
               myConnection.Close(); }
```

}

La imagen que mostramos enseguida corresponde al servicio web MensajesGrupo.asmx, dicho servicio web contiene un método con la finalidad es verificar si existen mensajes dentro del grupo de trabajo, toma como parámetros el Id del Grupo y retorna un valor boleano, true o false, true corresponde si hay mensajes y false en caso contrario, finalmente muestra los mensajes existentes en el grupo de trabajo (ver Figura 4.8).


Figura 4.8 Muestra los mensajes que se han creado dentro de un Grupo, en este ejemplo específico dentro lo grupo de Java.

```
using System.Web.Services;
using System.Configuration;
using System.Data.SqlClient;
using System.Data;
using System;

namespace Samples.Aspnet
{
   [WebService(Namespace = "http://tempuri.org/")]
   [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
   [System.Web.Script.Services.ScriptService]
   public class MensajesGrupo : System.Web.Services.WebService
   {
      protected SqlDataAdapter da;
      protected DataSet ds;
      private DataRow dr;
      protected SqlConnection myConnection = new SqlConnection();

[WebMethod(Description = " Metodo que checa si existen mensajes en un grupo determinado")]
      public bool ConsultaExistenMensajes(string idgrupo)
      {
         string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
         SqlConnection myConnection = new SqlConnection(strConnectionString);
         string strCommandText = "SELECT * FROM MensajesGrupo  WHERE idgrupo= '" + idgrupo
+ "' ORDER BY fechacreacion";
         da = new SqlDataAdapter(strCommandText, myConnection);
         ds = new DataSet();
```

```
        try {
            da.Fill(ds, "MensajesGrupo");
        }catch (Exception ex){
            Console.WriteLine(ex.Message);
        }
        finally {
            if (myConnection != null)
                myConnection.Close();
        }
        if (ds.Tables["MensajesGrupo"].Rows.Count > 0)
            return true;
        else
            return false;
}
```

una vez que se ha codificado los Servicios Web se procede a la compilación, en los siguentes ejemplos solo mostramos los resultados posteriores a la compilación del Servicio Web Grupos que describimos al inicio de los mismos.

La plataforma ASP .Net genera la descripción (WSDL) del servicio web a partir del código del Servicio Web Grupos.asmx (ver Figura 4.9).

```
<?xml version="1.0" encoding="utf-8" ?>
- <wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:tns="http://tempuri.org/"
    xmlns:s="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
    xmlns:http="http://schemas.xmlsoap.org/wsdl/http/" targetNamespace="http://tempuri.org/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  - <wsdl:types>
    - <s:schema elementFormDefault="qualified" targetNamespace="http://tempuri.org/">
      - <s:element name="ConsultaGrupos">
          <s:complexType />
        </s:element>
      - <s:element name="ConsultaGruposResponse">
        - <s:complexType>
          - <s:sequence>
            - <s:element minOccurs="0" maxOccurs="1" name="ConsultaGruposResult">
              - <s:complexType>
                - <s:sequence>
                    <s:any minOccurs="0" maxOccurs="unbounded"
                      namespace="http://www.w3.org/2001/XMLSchema" processContents="lax" />
                    <s:any minOccurs="1" namespace="urn:schemas-microsoft-com:xml-diffgram-v1"
                      processContents="lax" />
                  </s:sequence>
                </s:complexType>
              </s:element>
            </s:sequence>
          </s:complexType>
        </s:element>
      - <s:element name="ConsultaExisteGrupo">
        - <s:complexType>
          - <s:sequence>
              <s:element minOccurs="0" maxOccurs="1" name="grupo" type="s:string" />
            </s:sequence>
```

Figura 4.9 Muestra un fragmento de la Descripción del Servicio Web generado a partir de ASP .NET del Servicio Web Grupos.

Así como también el código referente a los mensajes SOAP encargados de la comunicación entre los servicios (ver Figura 4.10).

```
POST /Griposas/Grupos.asmx HTTP/1.1
Host: localhost
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-
envelope">
  <soap12:Body>
    <ConsultaGrupos xmlns="http://tempuri.org/" />
  </soap12:Body>
</soap12:Envelope>
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-
envelope">
  <soap12:Body>
    <ConsultaGruposResponse xmlns="http://tempuri.org/">
      <ConsultaGruposResult>xmlxml</ConsultaGruposResult>
    </ConsultaGruposResponse>
  </soap12:Body>
</soap12:Envelope>
```

Figura 4.10: muestra un ejemplo de solicitud y respuesta para SOAP 1.2 del método ConsultaGrupos del Servicio Web Grupos.asmx descrito anteriormente.

Este Servicio Web utiliza http://tempuri.org/ como espacio de nombres predeterminado que corresponde al UDDI local, para más información sobre UDDI consultar Capítulo 2 Cómputo Orientado a Servicios.

## 5. CONCLUSIÓN

Cada integrante de los grupos establecidos en el Sistema Apoyo al Aprendizaje Colaborativo (SISAAC) podrá acceder desde cualquier terminal remota sólo contando con una computadora y acceso a internet, lo cual ahorrará costos al estudiante hablando de tiempo y dinero, evitando la necesidad de trasladarse a la FCC para aclarar dudas, porque tendrá una red de compañeros conectados entre sí, compartiendo ideas, conocimientos, notas, referencias relevantes de algún tema en especial para reforzar los conocimientos adquiridos.

Las aportaciones principales del presente trabajo de tesis son:

- Desarrollo de un sistema que apoya al proceso del aprendizaje a través de la suma de esfuerzos colaborativos de la comunidad académica de la FCC.
- Desarrollo de sistemas basados en Servicios Web.
- Facilitar la integración de aplicaciones utilizando la tecnología de Servicios Web basados en SOA.
- Un entorno basados en grupos que apoye no sólo el aprendizaje colaborativo sino crear vínculos sociales entre los estudiantes de la FCC.

# Apéndices

# Apéndice A.
# Manual de Usuario.

SISAAC se encuentra integrado por dos sistemas principalmente que son el sistema administrador de usuarios y el entorno centrado en grupos de SISAAC, por tanto, en las siguientes secciones se presentan sus manuales respectivos.

## A.1. Sistema Administrador SISAAC.

Administración SISAAC es una aplicación dedicada para la gestión, administración de usuarios, a la cual solo el administrador tendrá facultad de hacer uso de ella, es decir sólo el administrador podrá actualizar, borrar, subir información, hacer alguna búsqueda etc., de usuarios SISAAC.
Para utilizar este sistema debe tener Instalado Visual Studio 2008, SQL SERVER 2008 con la base de datos de SISAAC previamente cargada y *Oficce* 2008 (Excel).
Comenzaremos mostrando la Interfaz gráfica antes de conectar a la base de datos (ver Figura A1), de la cual desglosaremos su contenido como botones, checkBox, etc. y describimos su función de cada uno de dichos elementos.
Al iniciar la aplicación nos mostrará una ventana inhabilitando la gran mayoría de elementos (botones, TextBox, etc.) como la que se muestra a continuación:

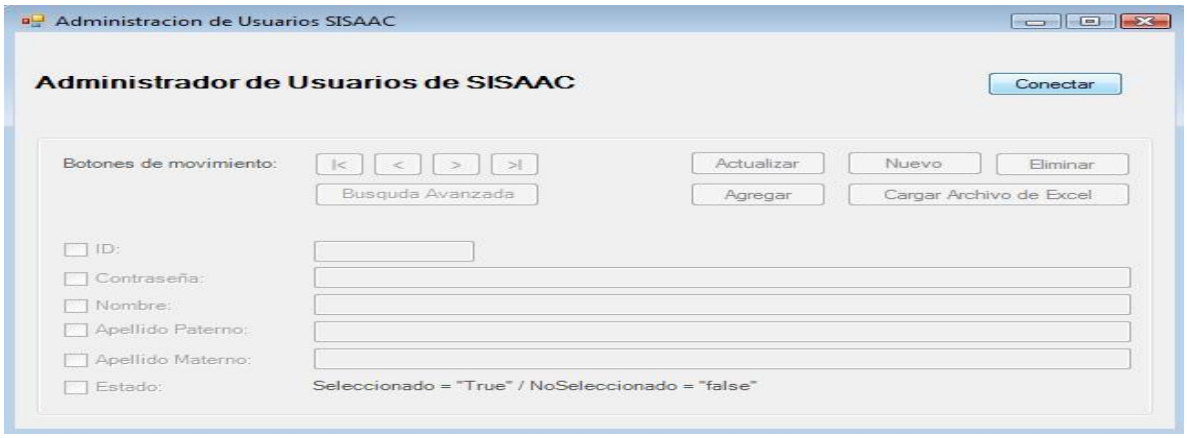



Figura A1: Interfaz Gráfica de Administrador SISAAC

Como podemos observar solo está habilitado el botón conectar y por supuesto los botones que se encuentran en la parte superior derecha que como ya es usual en la gran mayoría de ventanas Windows los incorpora, los cuales pertenecen a minimizar, restaurar y cerrar la ventana.
Para iniciar debemos primero comenzar con el botón conectar pues es muy importante porque habilita el resto de elementos del sistema.

**Botón Conectar:** La finalidad del botón conectar 

es realizar la conexión a la Base de Datos para poder así hacer consultas, eliminaciones, etc. de la

información de los usuarios SISAAC además para iniciar habilita todos los elementos de la interface (ver Figura A2).


Figura A2: Conectando a la base de datos de los usuarios y habilitando botones de la interfaz.

**Botones de Movimiento:**  La finalidad de los botones de movimiento es el desplazamiento de la información de la base de datos al inicio (botón inicial), al final (botón final) de la misma o ir desplazándose de un solo usuario hacia atrás (botón central izquierda) o al frente (botón central derecha). La información del usuario se muestra en la parte inferior de la interfaz, de a cuerdo al desplazamiento elegido por el administrador (ver Figura A3).

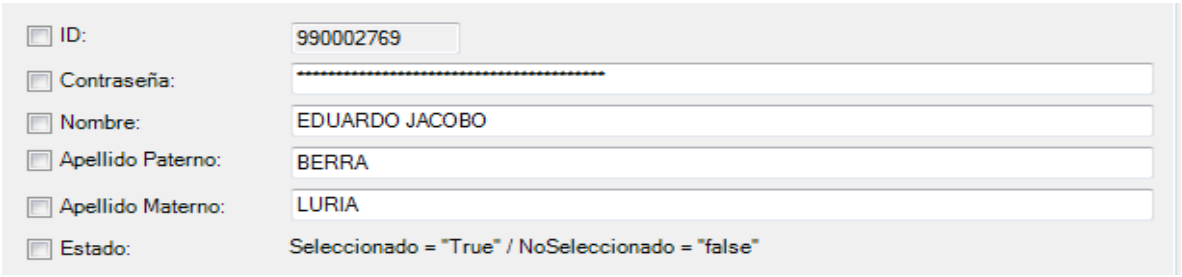
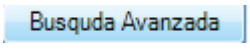



Figura A3: Ubicación de la interfaz exclusiva para mostrar información del usuario.

**Botón Búsqueda Avanzada:**  Abre una nueva ventana referente a la búsqueda del usuario de acuerdo a su ID que corresponde a su matrícula, en la Figura A4 mostramos un ejemplo el cual se está haciendo una consulta del usuario con ID 200310510 que con fines prácticos se ha tomado al azar, y en la Figura A5 se muestran los resultados de la búsqueda.

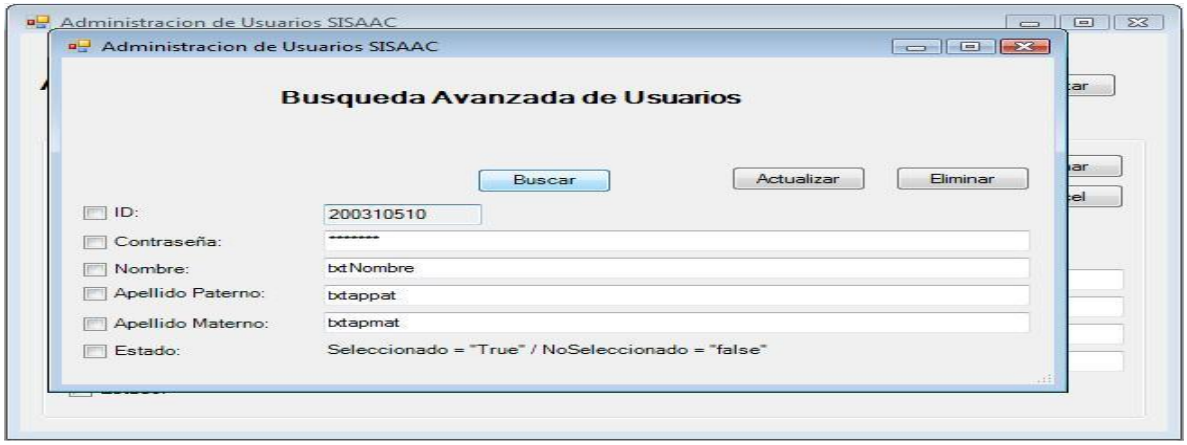
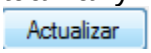



Figura A4: Ejemplo de búsqueda por Id de usuario 200310510


Figura A5: Muestra los resultados después de una búsqueda por Id de usuario.

Como podemos observar en la Figura A5 de la ventana de búsqueda muestra otros dos botones Actualizar y Eliminar los cuales se explican posteriormente.

**Botón Actualizar:** 

El botón actualizar tiene como finalidad hacer modificaciones a la información contenida del usuario seleccionado, para ello es conveniente mencionar que solo se harán las modificaciones si se selecciona los campos a modificar en los checkbox, enseguida mostramos un ejemplo el cual se va a modificar el apellido paterno y apellido materno del usuarios Patricia con ID 2009 40106 como se muestra en la Figura A6.


Figura A6: Información de usuario con ID 200940106 antes de hacer modificaciones.

Como se en la Figura A6 anterior tenemos al usuario con ID 200940106 sin hacer alguna modificación y en Figura A7 se muestra la información una vez actualizada, es decir en la Figura A6 el usuario Patricia tiene apellidos Montes Quiroz, después se muestra la información(ver Figura A7) del mismo usuario pero ahora con apellidos: Merino Gómez.


Figura A7: Muestra como se ha actualizado con éxito los datos del Usiario.

**El Botón Nuevo:** 

La función del Botón Nuevo es Añadir información de un nuevo usuario al a base de datos de usuarios SISAAC (dar de alta a un usuario), una vez seleccionado nos abrirá otra ventana permitiendo capturar los datos del usuario (ver Figura A8).

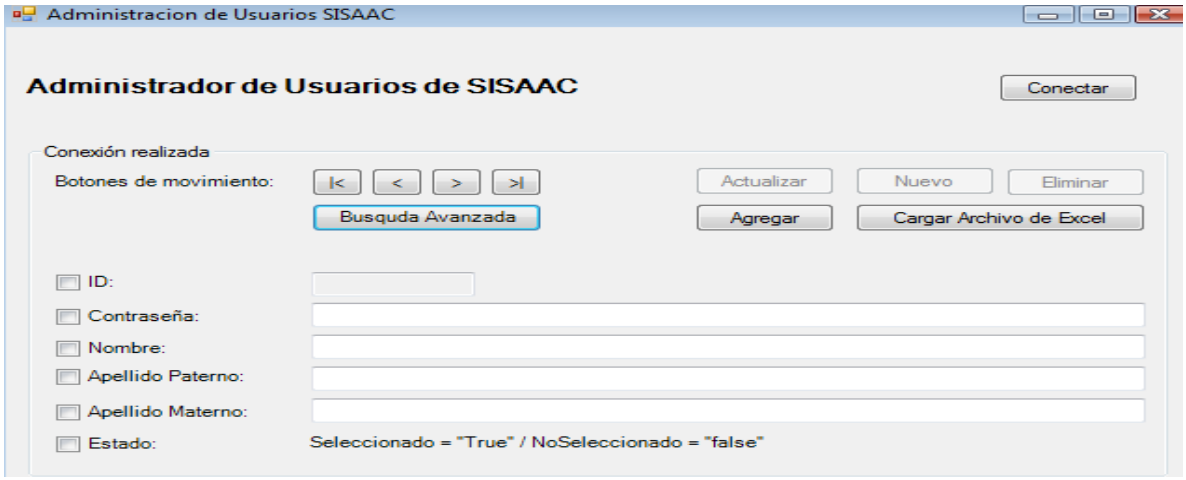
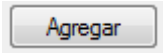



Figura A8 Ventana esperando los datos del Usuario a agregar.

Como ya hemos mencionado, para agregar la información debemos capturarla en el Formulario ubicado en la parte inferior de la venta, además tendremos que seleccionar todos los campos (chek Box) pues se agregará todos los datos del usuario. Una vez ingresados los datos damos clic al B**otón agregar** 

para decirle a la aplicación que guarde los cambios Figura A9.

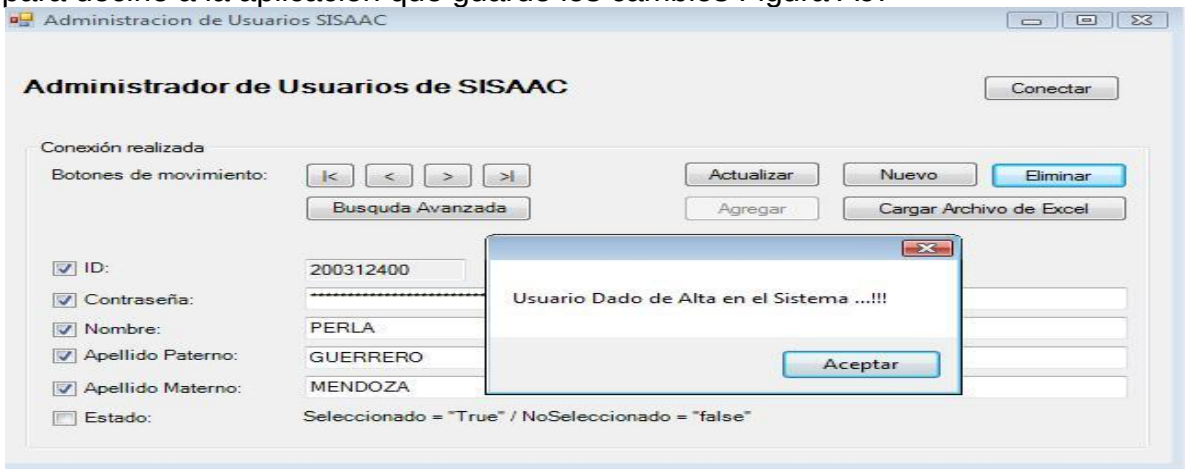
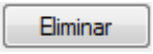



Figura A9: Muestra que un usuario se ha dado de alta.

Notas:

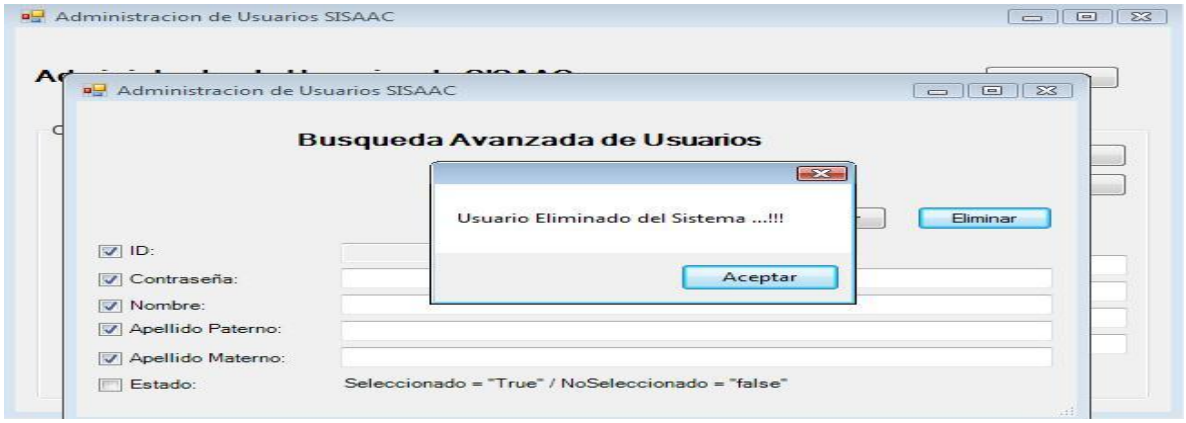
- El Chek Box estado: 

en este caso no se selecciona ya que es una bandera para determinar si se ha ingresado al sistema SISAAC , así que si activa la casilla, no permitirá al usuario hacer su registro. Por ello si es la primera vez que ha ingresado se deja en blanco.
- En el momento en que se ha agregado el usuario al sistema encripta las contraseñas automáticamente.

**Botón Eliminar:** 

Ahora si queremos eliminar algún usuario de la base de datos, localizamos el usuario a eliminar, seleccionamos los campos y oprimimos el botón eliminar, en la Figura A10 mostramos la información del

usuario 200312400 el cual agregamos en el ejemplo anterior de agregar, ahora lo eliminaremos (ver Figura A10).




FiguraA10: Muestra que el usuario con ID 200312400 ha sido eliminado del sistema.

**Botón Cargar Archivo de Excel:** 

El presente Botón tiene como finalidad agregar o dar de alta a varios usuarios a la vez, al dar clic en este botón nos abrirá otra nueva ventana como se muestra en la figura siguiente (ver Figura A11).

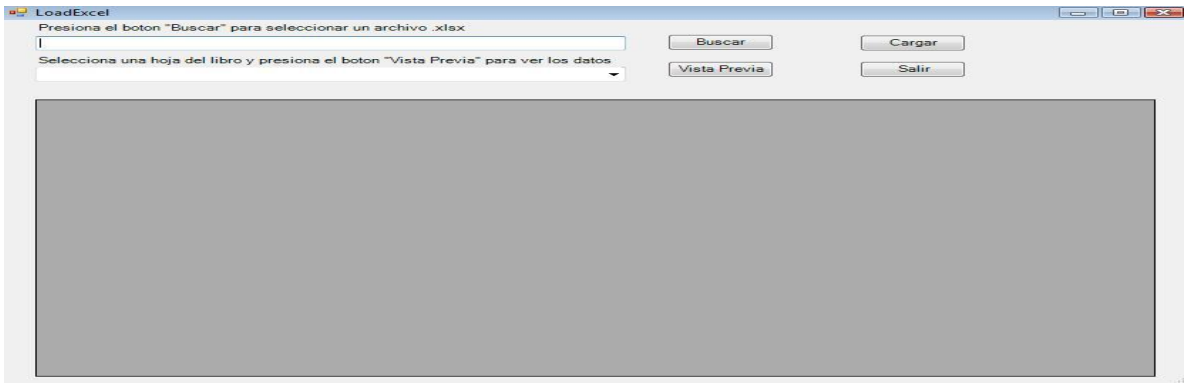
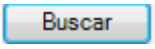
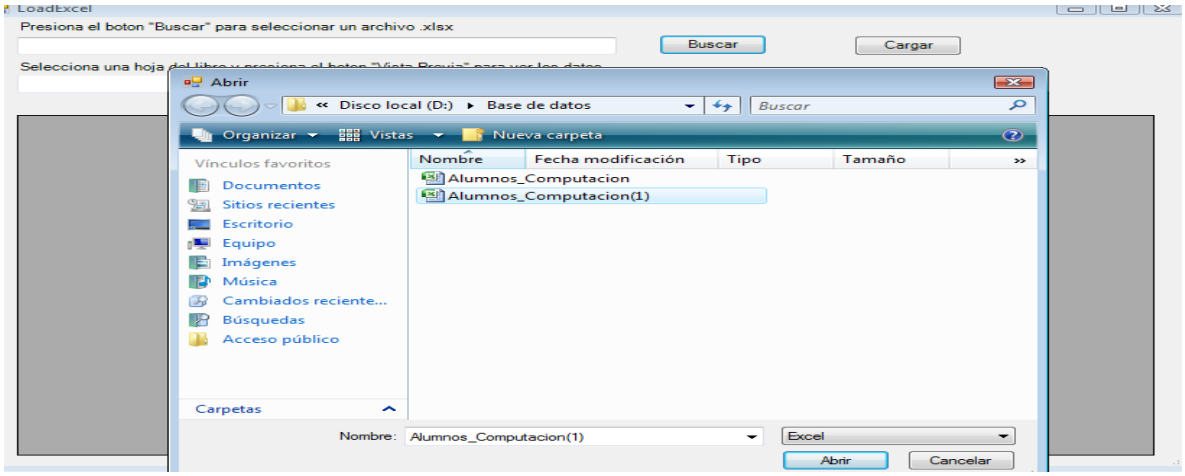



Figura A11: Muestra la ventana lista para agregar varios usuarios.

A continuación listaremos los pasos para importar los datos de varios usuarios a la base de datos que los contiene.

1. Al Dar clic al botón buscar, 

nos abrirá la ventana del explorador Windows para localizar el archivo en Excel que contiene los datos del Usuario, en nuestro caso elegiremos el Archivo llamado: Alumnos_Computación(1) y damos clic al botón abrir de la ventana del explorador de Windows (Ver Figura A12).




FiguraA12: Importando datos de usuarios.

2. Una vez abierto el archivo aparecerá la ruta o dirección del archivo seleccionado en la parte superior como se muestra en la Figura A13

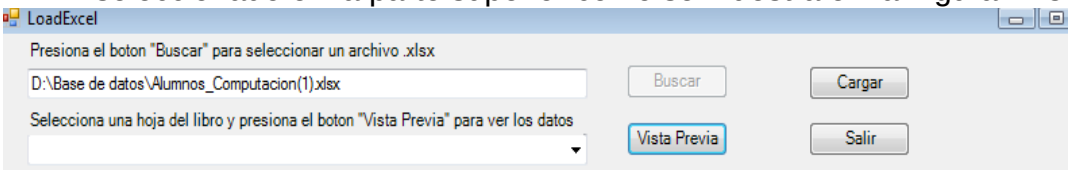



Figura A13: Muestra la dirección del archivo una vez abierto.

3. Hasta ahora llevamos abierto el archivo de Excel, pero no hemos seleccionado la hoja que contiene la información mencionada anteriormente, para ello desplegamos las opciones que se encuentran en la parte inferior de la Ruta del archivo dando clic en el triángulo pequeño al final de la caja de texto inferior  en nuestro ejemplo tenemos tres hojas, elegiremos la primera llamada FCC (ver Figura A14).

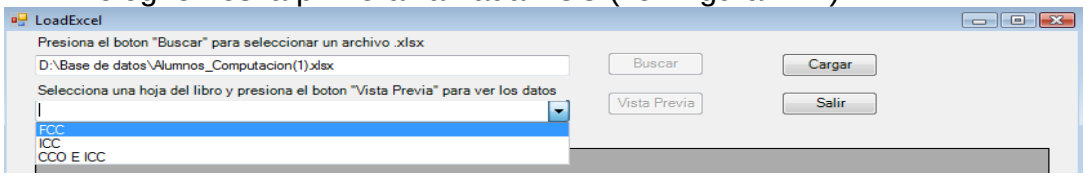



Figura A14: Despliega las hojas contenidas en el archivo Excel cargado.

4. Una vez elegida la hoja procedemos a oprimir el botón cagar  para importar los datos a la base de datos de usuarios SISAAC.
5. Para tener un panorama general de los datos que hemos importado tendremos que oprimir el botón Vista Previa, enseguida la aplicación mostrara los datos agregados.
6. Por último solo nos queda cerrar nuestra ventana pues los datos se han agregado.

NOTA: Al importar datos desde un archivo en Excel el administrador deberá encriptar las contraseñas.

Para el Encriptado de contraseñas de deberá abrir el sitio web (http://dominio /encriptar) y automáticamente la aplicación hace el encriptado de contraseñas.

### A.2. SISAAC (SIStema de Apoyo al Aprendizaje Colaborativo).

En este apartado se explica el uso de SISAAC por parte de los usuarios, para ingresar al sistema SISAAC el administrador debe realizar pre registro del usuario, el cual consta en dar de alta o ingresar información a la base de datos que contendrá todos los usuarios autorizados.
Cada usuario contará con ID de Usuario y contraseña, así que si es la primera vez que el usuario ingresa a SISAAC, por defecto su ID de usuario corresponde a su matrícula, la cual le asignaron en el momento que ingresó a la Universidad, y la contraseña corresponde de a los últimos 6 dígitos de la matrícula.
Por ejemplo: si el alumno Juan Pérez López con matrícula 200913856 quiere ingresar al sistema.

ID Usuario: 200913856
Contraseña: 913856

Para ser miembro de SISAAC debemos pasar por dos etapas:

1. Validar ID de Usuario y contraseña anteriormente descritos.
2. Completar registro.
3. Validar Datos: la validación consiste en ingresar el ID (matrícula) y contraseña (6 últimos dígitos de la matrícula) del usuarios, es sencillo ya que en la interfaz del usuario viene especificado con unas letras azules (ver Figura A15)


Figura A15: Interfaz Para validar ID de usuario y Contraseña.

## 1. Completar registro.

Una vez validado el ID de usuario y contraseña por defecto nos enviara al área de registro, en la cual deberá introducir los datos que se le piden, los campos son obligatorios, ningún campo se podría dejar en blanco, en la interfaz Grafica del usuario está etiquetado con letras negritas donde corresponden dichos datos (ver Figura A16).


Figura A16: Interfaz Grafica para el registro de usuario

Una vez completado el registro el podremos entrar a la Pantalla principal de SISAAC, la cual describiremos enseguida. Para usos prácticos dividiremos a la interfaz principal de SISAAC en tres partes: área donde se muestra el nombre de usuario que ha ingresado a SISAAC, barra de navegación y barra lateral (ver Figura A17)


Figura A17: Interfaz de la página principal SISAAC

En la barra lateral encontramos tres elementos cada uno de ellos con su nombre en la parte superior (elemento crear grupo, elemento calendario y elemento Todos los grupos).

El elemento **Crear grupo** como su nombre lo indica, permitirá al usuario crear un grupo de trabajo. Para crear un grupo debemos introducir en nombre del grupo y una pequeña descripción del tema que se abordará y por último haremos clic al botón crear Grupo (ver Figura A18).


Figura A18: Elemento para crear grupo.

Elemento **calendario,** dicho elemento es un calendario el cual muestra el mes actual.

El elemento **Todos los grupos** muestra todos los grupos que se han creado por los usuarios SISAAC, para ingresar solo basta seleccionar el grupo de nuestro interés y finalmente clic en el botón entrar (ver Figura A19).

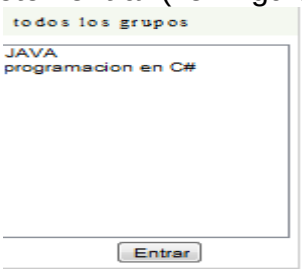



Figura A19: Muestra todos los grupos creados en SISAAC

Al ingresar a algún Grupo de trabajo podemos enviar mensajes entre usuarios, con el fin de aportar ideas, hacer un comentario respecto al grupo, etc.
En el ejemplo de la Figura A20 hemos ingresado al grupo de C#, y se muestra el área para escribir el mansaje, así como el área donde se muestra el mensaje escrito por el usuario


Figura A20: Mandar Mensajes entre usuarios dentro del grupo de C#

En la Barra de Navegación (ver Figura A21) nos permite trasladarnos a cada sección de SISAAC de acuerdo a la opción hecha por el usuario.


Figura A21: Barra de navegación de SISAAC

Dentro de la barra de navegación tenemos siete opciones que elegir (HOME, MI CUENTA, BUAP, FCC, A CERCA DE, MI BLOG, CERRAR SESION) que explicaremos a continuación.

- **Home:** 

Es un vínculo a la página principal de SISAAC.
- **BUAP** 

**:** Hace un vínculo a la página Oficial de la BUAP (Benemérita Universidad Autónoma de Puebla) y la cual permite ingresar a ella.

- **Bibliotecas BUAP** 

**:** la presente opción hace referencia y permite ingresar a la agina oficial de Bibliotecas de la BUAP
- **FCC** 

**:** Es un vínculo a la página oficial de la Facultad de Ciencias de la Computación.
- **Acerca de** 

**:** la finalidad es mostrar los objetivos e información de los colaboradores que hicieron posible la implementación de SISAAC.
- **Mi Blog** 

**:** Nos abre otra ventana correspondiente a la interfaz del Blog de cada usuario, dicha herramienta es descrita a detalle en el manual de usuario que corresponde a la tesis: Herramientas Colaborativas basadas en Servicios Web para una Plataforma de Apoyo al Aprendizaje en la FCC.
- **Cerrar sesión** 

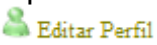
**:** como su nombre lo indica cierra la sesión del usuario es decir salir de SISAAC, y envía a la página de validación de usuario.

Y por último explicaremos el vínculo **MI CUENTA** 

**,** esta opción es el panel de control del usuario, ya que puede cambiar su información básica como son contraseña, perfil, unirse a más grupos de abajo y cambiar su imagen de perfil (ver Figura A22).


Figura A22: Panel de control de SISAAC.

Para hacer cambios a los datos que introducimos en nuestro registro, presentamos las siguientes opciones que se encuentran en la barra lateral izquierda.

**Editar Perfil** 

Muestra la información que introducimos al hacer el registro del usuario para ingresar a SISAAC, además nos permite cambiar el correo y nuestro mensaje personal (ver Figura A24).


Figura A23: Interfaz para editar el perfil del usuario

En **Editar correo y contraseña:** Editar correo y contraseña podemos cambiar el correo y contraseña que inicialmente ingresamos en el registro de usuario. Para cambiar el correo o contraseña primero debemos ingresar nuestra contraseña actual, esto se hace por seguridad, impidiendo que terceros usuarios puedan hacer mal uso de la cuenta, por ello es que se inhabilita las dos secciones finales.
Una vez ingresado la contraseña actual y sea correcta, se habilitaran las secciones para editar el correo y contraseña, esperando que sean ingresados los nuevos datos, finalmente oprimimos el botón guardar para que se guarden los cambios hechos (Ver Figura A24).

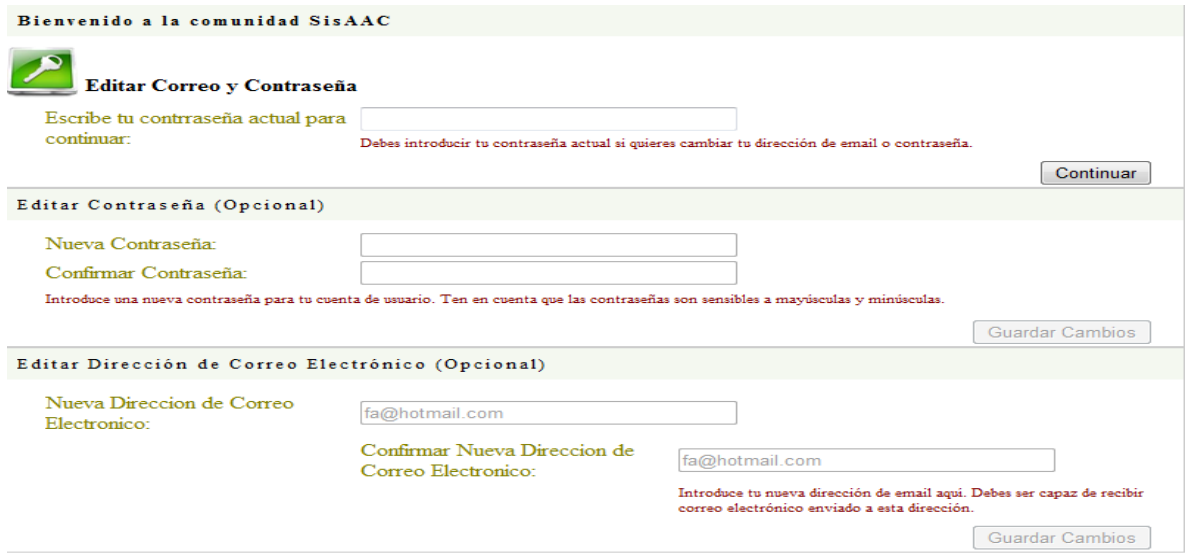



Figura A24 Interfaz para editar correo y contraseña

En la opción **Cambiar imagen del Perfil** Cambiar la Imagen de mi Perfil:,es la interfaz que hace posible cambiar la imagen del perfil de usuario que inicialmente subimos a SISAAC en el formulario de registro(ver Figura A25).


Figura A25: Interfaz para cambiar la imagen del perfil de usuario

El botón examinar permite que localicemos la imagen a través de la ventana del explorador de Windows, una vez localizada damos clic al botón cambiar imagen para guardar los cambios en el SIAAC.

Por último concluimos con la opción **Grupos** 

**,** para que un usuario pueda pertenecer a más grupos, como podemos observar en la Figura A26 muestra la interfaz de los grupos de trabajo, en la parte izquierda se listan todos los grupos que se han creado en el entorno SISAAC y en la parte derecha los grupos a los que pertenece el usuario.

Para unirse a un grupo seleccionamos el grupo al que me interese unirme y enseguida dar clic al botón agregar, por el contrario y ya no quiero pertenecer a algún grupo, seleccionar el grupo de trabajo dentro de mis grupos y dar clic al Botón eliminar.

Solo se podrá eliminar a algún grupo si pertenece a dicho grupo.


Figura A26: Muestra el interfaz para unirse o dejar un grupo de trabajo.

Veremos que contamos con más opciones en la barra lateral que no fueron descritas porque pertenecen a en otro trabajo de tesis: Herramientas Colaborativas basadas en Servicios Web para una Plataforma de Apoyo al Aprendizaje en la FCC (Facultad de Ciencias de la Computación), en la cual se describen a detalle.

# APÉNDICE B.
# MANUAL TÉCNICO.

El presente manual técnico tiene como finalidad mostrar el código desarrollado para SISAAC, así como del sistema Administrador de usuarios SISAAC, además dar a conocer las herramientas, lenguajes de programación y gestor de Bases de Datos involucrados para su implementación.

El **sistema administrador de SISAAC** es una aplicación de escritorio, desarrollado con Windows Forms con el lenguaje de programación C#, bajo la plataforma .NET, y el gestor de Bases de Datos SQL SERVER 2008.

El archivo inicio.cs ejecuta, carga la interfaz de inico de la aplicación, se encarga de la conexión a la base de datos donde se encuentra contenida la información de los usuarios, inhabilita/habilita los Text Box, Check Box y botones, asi como de invocar los metodos que se encuentran contenidas dentro del mismo archivo para realizar las operaciones requeridas.

incicio.cs

```
using System;
using System.Windows.Forms;
using System.Drawing;
using System.Data;
using System.Data.SqlClient;
using System.Security.Cryptography;
using System.Text;
   public class Form1 : System.Windows.Forms.Form{
      [STAThread]
      static void Main(){
 Application.Run(new Form1());
      }
      public Form1(){
         Application.EnableVisualStyles();
         Application.DoEvents();
         InitializeComponent();
      }
      private DataTable dt;
      private SqlDataAdapter da;
      private int fila;
      private void Form1_Load(object sender, EventArgs e) {
         this.Text = "Administracion de Usuarios SISAAC";
         foreach (Control c in this.GroupBox1.Controls) {
            if (c is TextBox) { c.Text = ""; }
            c.Enabled = false;
         }
         this.GroupBox1.Enabled = false;
      }
      private void btnConectar_Click(object sender, EventArgs e) {
         string sCnn = "Server=(local); database=sisaac; integrated security=yes";
         SqlConnection cnn = new SqlConnection(sCnn);
         string sSel = "SELECT * FROM Usuarios ORDER BY IdUsuario DESC";
         try {
```

```
            da = new SqlDataAdapter(sSel, sCnn);
            SqlCommandBuilder cb = new SqlCommandBuilder(da);
            da.UpdateCommand = cb.GetUpdateCommand();
            da.InsertCommand = cb.GetInsertCommand();
            da.DeleteCommand = cb.GetDeleteCommand();
            da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
            dt = new DataTable();
            da.Fill(dt);
            foreach (Control c in this.GroupBox1.Controls) {
                c.Enabled = true;
            }
            btnNuevo.Enabled = false;
            this.GroupBox1.Enabled = true;
            this.GroupBox1.Text = "Conexión realizada";
            if (dt.Rows.Count > 0) {
                btnFirst_Click(null, null);
            }
            else{
                fila = -1;
                btnActualizar.Enabled = false;
            }
        }
        catch (Exception ex) {
            MessageBox.Show("ERROR al conectar o recuperar los datos:\n" +
                ex.Message, "Conectar con la base",
                MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
    private void asignarDatos(DataRow dr)  {
        if(checkBox1.Checked)
            dr["idusuario"] = txtIdUsuario.Text;
        dr["nusuario"] = txtNombre.Text;
        if (checkBox2.Checked)
            dr["pswd"] = generarClaveSHA1(txtpswd.Text);
        if (checkBox3.Checked)
            dr["nombre"] = txtNombre.Text;
        if (checkBox4.Checked)
            dr["appat"] = txtappat.Text;
        if (checkBox5.Checked)
            dr["apmat"] = txtapmat.Text;
        if (!checkBox6.Checked)
            dr["estado"] = "false";
        else
            dr["estado"] = "truee";
    }
    private void mostrarDatos(int f) {
        int uf = dt.Rows.Count - 1;
        if (f < 0 || uf < 0) return;
        DataRow dr = dt.Rows[f];
        txtIdUsuario.Text = dr["idusuario"].ToString();
        txtpswd.Text = dr["pswd"].ToString();
        txtNombre.Text = dr["Nombre"].ToString();
        txtappat.Text = dr["appat"].ToString();
        txtapmat.Text = dr["apmat"].ToString();
        btnActualizar.Enabled = true;
        btnEliminar.Enabled = true;
```

```
}
private void btnFirst_Click(object sender, EventArgs e)  {
    fila = 0;
    mostrarDatos(fila);
}
private void btnPrev_Click(object sender, EventArgs e) {
    fila = fila - 1;
    if (fila < 0) fila = 0;
    mostrarDatos(fila);
}
private void btnNext_Click(object sender, EventArgs e)  {
    int uf = dt.Rows.Count - 1;
    fila = fila + 1;
    if (fila > uf) fila = uf;
    mostrarDatos(fila);
}
private void btnLast_Click(object sender, EventArgs e) {
    fila = dt.Rows.Count - 1;
    mostrarDatos(fila);
}
private void btnActualizar_Click(object sender, EventArgs e)
{
    if (fila < 0 || fila > dt.Rows.Count - 1) return;
    DataRow dr = dt.Rows[fila];
    asignarDatos(dr);
    try{
        da.Update(dt);
        dt.AcceptChanges();
        MessageBox.Show("Datos del Usuario Actualizados ...!!!");
    }
    catch (DBConcurrencyException ex) {
        MessageBox.Show("Error de concurrencia:\n" + ex.Message);
    }
    catch (Exception ex) { MessageBox.Show(ex.Message); }
}
private void btnNuevo_Click(object sender, EventArgs e)
{
    DataRow dr = dt.NewRow();
    asignarDatos(dr);
    dt.Rows.Add(dr);
    try{
        da.Update(dt);
        dt.AcceptChanges();
        if (Convert.ToInt32("0" + dr["idusuario"].ToString()) == 0) {
            dt = new DataTable();
            da.Fill(dt);
        }
        btnLast_Click(null, null);
    }
    catch (DBConcurrencyException ex) {
        MessageBox.Show("Error de concurrencia:\n" + ex.Message);
    }
    catch (Exception ex) { MessageBox.Show(ex.Message);  }
    btnNuevo.Enabled = false;
    btnlimpiar.Enabled = true;
    btnActualizar.Enabled = true;
```

```
            btnEliminar.Enabled = true;
            btnExcel.Enabled = true;
            MessageBox.Show("Usuario Dado de Alta en el Sistema ...!!!");
        }
        private void btnEliminar_Click(object sender, EventArgs e) {
            if (fila < 0 || fila > dt.Rows.Count - 1) return;
            try{
                dt.Rows[fila].Delete();
                da.Update(dt);
                dt.AcceptChanges();
                txtIdUsuario.Text = txtpswd.Text = txtNombre.Text = txtappat.Text = txtapmat.Text = "";
                MessageBox.Show("Usuario Eliminado del Sistema ...!!!");
            }
            catch (DBConcurrencyException ex) {
                MessageBox.Show("Error de concurrencia:\n" + ex.Message);
            }
            catch (Exception ex) {
                MessageBox.Show(ex.Message);
            }
        }
        private void button1_Click_1(object sender, EventArgs e) {
            Form LoadExcel = new SISAAC.LoadExcel();
            LoadExcel.ShowDialog();
        }
        private void button2_Click(object sender, EventArgs e) {
            txtIdUsuario.Text = txtpswd.Text = txtNombre.Text = txtappat.Text = txtapmat.Text = "";
            btnNuevo.Enabled = true;
            btnlimpiar.Enabled = false;
            btnActualizar.Enabled = false;
            btnEliminar.Enabled = false;
        }
        private void button1_Click_2(object sender, EventArgs e) {
            Form Buscar = new SISAAC.Buscar();
            Buscar.ShowDialog();
        }
        public string generarClaveSHA1(string pass) {
            UTF8Encoding enc = new UTF8Encoding();
            byte[] data = enc.GetBytes(pass + "sisaac-fcc-buap");
            byte[] result;
            SHA1CryptoServiceProvider sha = new SHA1CryptoServiceProvider();
            result = sha.ComputeHash(data);
            StringBuilder sb = new StringBuilder();
            for (int i = 0; i < result.Length; i++){
                if (result[i] < 16) {  sb.Append("0");}
                sb.Append(result[i].ToString("x"));
            }
            return sb.ToString().ToUpper();
        }
    }
```

**SISAAC** se desarrolló en ASP.NET y C#.NET (para servicios web), por la gama de herramientas que el Framework .NET provee al programador, además se hizo uso de herramientas externas del Framework.NET, como son CSS, JavaScript y DOM las cuales tienen la facilidad de trabajar con el Framework .NET para darle una mejor funcionalidad (dinámica) y apariencia (amigable) a las aplicaciones Web y SQL SERVER 2008 (ADO.NET) para la base de datos. Es importante hacer

mención que **Erwin** proporciona la facilidad de generar los scripts para generar las tablas de la base de datos, es por ello que se hizo uso de dicha herramienta**.**

SISAAC se divide en cinco secciones: Inicio de sesión, Registro de usuario, Entorno principal, Grupos y Panel de control. Posteriormente se presentará el código de JavaScript, CSS y Base de datos.

### Inicio de Sesión

El Servicio Web AccessWS.asmx cuenta con los métodos: SearchUser y Estado_usuario, y es el encargado de verificar si el usuario existe en la BD así como también el estado del usuario, es decir, revisa si el usuario ha ingresado anteriormente a SISAAC.

AccessWS.asmx

```
using System.Web.Services;
using System.Configuration;
using System.Data.SqlClient;
using System.Data;
using System;

[WebService(Namespace = "http://microsoft.com/webservices/", Description = "Acceso a la base de datos Alumos (local) desde un servicio Web")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
System.Web.Script.Services.ScriptService
public class AccessWS : System.Web.Services.WebService{
    private SqlDataAdapter da;
    private DataSet ds;
    private SqlConnection myConnection = new SqlConnection();

    [WebMethod(Description = " Metodo que retorna un mensaje de texto indicando si se encontrado el registro buscado")]
    public bool SearchUser(string id,string pass){
        string strConnectionString = ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
        SqlConnection myConnection = new SqlConnection(strConnectionString);
        string strCommandText = "SELECT IdUsuario,pswd FROM Usuarios WHERE IdUsuario = '"+id+"' AND pswd = '"+pass+"'";
        da = new SqlDataAdapter(strCommandText, myConnection);
        ds = new DataSet();
        try { da.Fill(ds, "Usuarios"); }
        catch (Exception ex) { Console.WriteLine(ex.Message); }
        finally {
            if (myConnection != null)
                myConnection.Close();
        }
        if (ds.Tables["Usuarios"].Rows.Count > 0)
            return true;
        else
            return false;
    }
```

```
   [WebMethod(Description = " Metodo que retorna el estado del usuario detro del DB")]
   public string[] Estado_Usuario(string id) {
      string                                    strConnectionString                                    =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
      SqlConnection myConnection = new SqlConnection(strConnectionString);
      string strCommandText = "SELECT estado,nusuario FROM Usuarios WHERE IdUsuario = '" +
id +"'";
      da = new SqlDataAdapter(strCommandText, myConnection);
      ds = new DataSet();
      string[] estado = new string[2];
      try{   da.Fill(ds, "Usuarios"); }
      catch (Exception ex)   {
         Console.WriteLine(ex.Message);
         return null;
      }
      finally   {
         if (myConnection != null)
            myConnection.Close();
      }
      estado[0]=(ds.Tables["Usuarios"].Rows[0][0]).ToString ();
      estado[1]=(ds.Tables["Usuarios"].Rows[0][1]).ToString ();
      return estado;
   }
}
```

**Registro de usuario.**

El Servicio Web Perfil.asmx que contiene los métodos: consultaPerfil, Actualizar_generales, Actualiza_Pswd, Asignar_Datos, ActualizarDatos, get_mail, check_identidad, check_nusuario, Actualizar_email, Retornar_Usuarios, get_idusuario, get_nusuario, get_nomcom, get_imagen y Actualizat_imagen , se encarga de almacenar los datos capturados en el formulario de registro en la base de datos de usuarios SISAAC.

Perfil.asmx

```
using System.Web.Services;
using System.Configuration;
using System.Data.SqlClient;
using System.Data;
using System;

namespace Samples.Aspnet{
   [WebService(Namespace = "http://tempuri.org/")]
   [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
   [System.Web.Script.Services.ScriptService]
   public class Perfil : System.Web.Services.WebService{
      protected SqlDataAdapter da;
      protected DataSet ds;
      protected SqlConnection myConnection = new SqlConnection();

      public Perfil()  { }

      [WebMethod(Description = " Metodo que actualiza el comentario de un Usuario")]
```

```
        public void Actualizar_Generales(string id, string comentario) {
            string strConnectionString = ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
            SqlConnection myConnection = new SqlConnection(strConnectionString);
            SqlCommand cmd = new SqlCommand();
            cmd.CommandType = CommandType.Text;
            cmd.CommandText = "UPDATE usuarios SET comentario = '" + comentario + "' " +
                        "WHERE IdUsuario = '" + id + "'";
            cmd.Connection = myConnection;
            myConnection.Open();
            cmd.ExecuteNonQuery();
            myConnection.Close();
        }

        [WebMethod(Description = " Metodo que actualiza el password de un Usuario")]
        public void Actualizar_Pswd(string id, string new_pass) {
            string strConnectionString = ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
            SqlConnection myConnection = new SqlConnection(strConnectionString);
            SqlCommand cmd = new SqlCommand();
            cmd.CommandType = CommandType.Text;
            cmd.CommandText = "UPDATE usuarios SET pswd = '" + new_pass + "' " + "WHERE IdUsuario = '" + id + "'";
            cmd.Connection = myConnection;
            myConnection.Open();
            cmd.ExecuteNonQuery();
            myConnection.Close();
        }

        [WebMethod(Description = " Metodo que retorna un datatable con el perfil de un Usuario")]
        public DataTable consultaPerfil(string id) {
            string strConnectionString = ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
            SqlConnection myConnection = new SqlConnection(strConnectionString);
            string strCommandText = "SELECT * FROM Usuarios " + " WHERE IdUsuario = '" + id + "'";
            da = new SqlDataAdapter(strCommandText, myConnection);
            SqlCommandBuilder cb = new SqlCommandBuilder(da);
            da.UpdateCommand = cb.GetUpdateCommand();
            da.InsertCommand = cb.GetInsertCommand();
            da.DeleteCommand = cb.GetDeleteCommand();
            da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
            DataTable dt = new DataTable();
            try{da.Fill(dt); }
            catch (Exception ex)  {
                Console.WriteLine(ex.Message);
                return null;
            }
            finally{
                if (myConnection != null)
                    myConnection.Close();
            }
            return dt;
        }

        [WebMethod(Description = " Metodo que actualiza el perfil de un Usuario retornando una cadena con un mesaje de acuerdo a la operacion")]
```

```
        public string ActualizarDatos(string id, string[] datos) {
            string                                          strConnectionString                                          =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
            SqlConnection myConnection = new SqlConnection(strConnectionString);
            string strCommandText = "SELECT * FROM Usuarios " + "WHERE IdUsuario = '" + id + "'";
            da = new SqlDataAdapter(strCommandText, myConnection);
            SqlCommandBuilder cb = new SqlCommandBuilder(da);
            da.UpdateCommand = cb.GetUpdateCommand();
            da.InsertCommand = cb.GetInsertCommand();
            da.DeleteCommand = cb.GetDeleteCommand();
            da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
            DataTable dt = new DataTable();
            try{ da.Fill(dt); }
            catch (Exception ex) {
                Console.Write(ex.Message);
                return "Error al conectarse a la DB";
            }
            try{
                DataRow dr = dt.Rows[0];
                asignar_datos(dr, datos);
                da.Update(dt);
                dt.AcceptChanges();
            }
            catch (DBConcurrencyException ex) {
                Console.WriteLine(ex.Message);
                return "Informacion no actualizada";
            }
            finally{
                if (myConnection != null)
                    myConnection.Close();
            }
            return "Informacion Actualizada";
        }

        private void asignar_datos(DataRow dr, string[] datos)  {
            dr["nusuario"] = datos[0];
            dr["pswd"] = datos[1];
            dr["nombre"] = datos[2];
            dr["appat"] = datos[3];
            dr["apmat"] = datos[4];
            dr["genero"] = datos[5];
            dr["fechan"] = datos[6];
            dr["email"] = datos[7];
            dr["facultad"] = datos[8];
            dr["carrera"] = datos[9];
            dr["campus"] = datos[10];
            dr["comentario"] = datos[11];
            dr["imagen"]=datos[12];
            dr["estado"] = "truee";
        }

        [WebMethod(Description = " Metodo que retorna el email de un usuario")]
        public string get_email(string idusuario) {
            string                                          strConnectionString                                          =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
            SqlConnection myConnection = new SqlConnection(strConnectionString);
```

```
            string strCommandText = "SELECT email FROM usuarios where idusuario='"+idusuario+"'";
            da = new SqlDataAdapter(strCommandText, myConnection);
            SqlCommandBuilder cb = new SqlCommandBuilder(da);
            da.InsertCommand = cb.GetInsertCommand();
            da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
            DataTable dt = new DataTable();
            string mensaje = "";
            try{da.Fill(dt); }
            catch (Exception ex) {
                Console.WriteLine(ex.Message);
                return null;
            }
            finally{
                if (myConnection != null)
                    myConnection.Close();
            }
            if (dt.Rows.Count > 0)
                mensaje = dt.Rows[0][0].ToString();
            return mensaje;
        }

        [WebMethod(Description = " Metodo que retorna un bool  a partir de la busqueda de un
password en la db")]
        public Boolean check_identidad(string idusuario, string pswd) {
            string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
            SqlConnection myConnection = new SqlConnection(strConnectionString);
            string strCommandText = "SELECT * FROM usuarios where idusuario='" + idusuario + "' and
pswd='" + pswd + "'";
            da = new SqlDataAdapter(strCommandText, myConnection);
            SqlCommandBuilder cb = new SqlCommandBuilder(da);
            da.InsertCommand = cb.GetInsertCommand();
            da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
            DataTable dt = new DataTable();
            Boolean mensaje = false;
            try{da.Fill(dt);   }
            catch (Exception ex) {
                Console.WriteLine(ex.Message);
                return false;
            }
            finally{
                if (myConnection != null)
                    myConnection.Close();
            }
            if (dt.Rows.Count > 0)
                mensaje = true;
            return mensaje;
        }

[WebMethod(Description="Metodo que retorna un true si el nombre de usuario ya existe en la DB")]
        public Boolean check_nusuario(string nusuario)  {
            string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
            SqlConnection myConnection = new SqlConnection(strConnectionString);
           string strCommandText="SELECT nusuario FROM usuarios where nusuario='"+nusuario+"'";
            da = new SqlDataAdapter(strCommandText, myConnection);
```

```
            SqlCommandBuilder cb = new SqlCommandBuilder(da);
            da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
            DataTable dt = new DataTable();
            Boolean mensaje = false;
            try{   da.Fill(dt); }
            catch (Exception ex) {
                Console.WriteLine(ex.Message);
                return false;
            }
            finally{
                if (myConnection != null)
                    myConnection.Close();
            }
            if (dt.Rows.Count > 0)
                mensaje = true;
            else
                mensaje = false;
            return mensaje;
        }

        [WebMethod(Description = " Metodo que actualiza el email de un usuario")]
        public void Actualizar_email(string id, string new_email) {
            string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
            SqlConnection myConnection = new SqlConnection(strConnectionString);
            SqlCommand cmd = new SqlCommand();
            cmd.CommandType = CommandType.Text;
cmd.CommandText="UPDATE usuarios SET email='"+new_email+"'"+"WHERE IdUsuario='"+id+"'";
            cmd.Connection = myConnection;
            myConnection.Open();
            cmd.ExecuteNonQuery();
            myConnection.Close();
        }

        [WebMethod(Description = " Metodo que retorna un datatable con los nombres de todos los
Usuarios registrados")]
        public DataTable RetornaUsuarios(string id) {
            string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
            SqlConnection myConnection = new SqlConnection(strConnectionString);
         string strCommandText="SELECT nusuario FROM Usuarios "+" WHERE IdUsuario!='"+id+ "'"
                                +"AND estado = 'truee' ORDER BY nusuario";
            da = new SqlDataAdapter(strCommandText, myConnection);
            SqlCommandBuilder cb = new SqlCommandBuilder(da);
            da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
            DataTable dt = new DataTable();
            try{da.Fill(dt); }
            catch (Exception ex) {Console.WriteLine(ex.Message); }
            finally{
                if (myConnection != null)
                    myConnection.Close();
            }
            return dt;
        }

        [WebMethod(Description = " Metodo que retorna el ID de un usuario a partir de su nombre")]
```

```
        public string get_idusuario(string nusuario) {
            string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
            SqlConnection myConnection = new SqlConnection(strConnectionString);
        string strCommandText = "SELECT idusuario FROM usuarios where nusuario='"+nusuario+"'";
            da = new SqlDataAdapter(strCommandText, myConnection);
            SqlCommandBuilder cb = new SqlCommandBuilder(da);
            da.InsertCommand = cb.GetInsertCommand();
            da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
            DataTable dt = new DataTable();
            string mensaje = "";
            try{da.Fill(dt); }
            catch (Exception ex) {
                Console.WriteLine(ex.Message);
                return null;
            }
            finally{
                if (myConnection != null)
                    myConnection.Close();
            }
            if (dt.Rows.Count > 0)
                mensaje = dt.Rows[0][0].ToString();
            return mensaje;
        }

        [WebMethod(Description = " Metodo que retorna el nombre de un usuario a partir de su ID")]
        public string get_nusuario(string id) {
            string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
            SqlConnection myConnection = new SqlConnection(strConnectionString);
            string strCommandText = "SELECT nusuario FROM usuarios where idusuario='" + id + "'";
            da = new SqlDataAdapter(strCommandText, myConnection);
            SqlCommandBuilder cb = new SqlCommandBuilder(da);
            da.InsertCommand = cb.GetInsertCommand();
            da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
            DataTable dt = new DataTable();
            string mensaje = "";
try{da.Fill(dt); }
            catch (Exception ex) {
                Console.WriteLine(ex.Message);
                return null;
            }
            finally{
                if (myConnection != null)
                    myConnection.Close();
            }
            if (dt.Rows.Count > 0)
                mensaje = dt.Rows[0][0].ToString();
            return mensaje;
        }

        [WebMethod(Description = " Metodo que retorna el nombre y el comentario de un usuario a
partir de su ID")]
        public string[] get_nomcom(string id) {
            string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
```

```
        SqlConnection myConnection = new SqlConnection(strConnectionString);
        string strCommandText = "SELECT nombre, appat, comentario, imagen, nusuario, fechan,
genero, carrera FROM usuarios where idusuario='" + id + "'";
        da = new SqlDataAdapter(strCommandText, myConnection);
        SqlCommandBuilder cb = new SqlCommandBuilder(da);
        da.InsertCommand = cb.GetInsertCommand();
        da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
        DataTable dt = new DataTable();
        try{da.Fill(dt); }
        catch (Exception ex) {
            Console.WriteLine(ex.Message);
            return null;
        }
        finally{
            if (myConnection != null)
                myConnection.Close();
        }
        string[] mensaje = new string[7];
        if (dt.Rows.Count > 0) {
            mensaje[0] = dt.Rows[0][0].ToString()+" "+dt.Rows[0][1].ToString();
            mensaje[1] = dt.Rows[0][2].ToString();
            mensaje[2]=  "../SISAAC/profile/imgUsers/"+dt.Rows[0][3].ToString();
            mensaje[3]=dt.Rows[0][4].ToString();
            mensaje[4]=dt.Rows[0][5].ToString();
            mensaje[5]=dt.Rows[0][6].ToString();
            mensaje[6]=dt.Rows[0][7].ToString();
        }
        else
            mensaje = null;
        return mensaje;
    }

    [WebMethod(Description = " Metodo que retorna la imagen de un usuario a partir del id de
usuario")]
    public string[] get_imagen(string id) {
        string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
        SqlConnection myConnection = new SqlConnection(strConnectionString);
        string strCommandText = "SELECT imagen FROM usuarios where idusuario='" + id + "'";
        da = new SqlDataAdapter(strCommandText, myConnection);
        SqlCommandBuilder cb = new SqlCommandBuilder(da);
        da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
        DataTable dt = new DataTable();
        try{da.Fill(dt); }
        catch (Exception ex) {
            Console.WriteLine(ex.Message);
        }
        finally{
            if (myConnection != null)
                myConnection.Close();
        }
        string[] mensaje = new string[2];
        if (dt.Rows.Count > 0) {
            mensaje[0] = "..\\sisaac\\profile\\imgUsers\\"+dt.Rows[0][0].ToString();
            mensaje[1] = @"c:\inetpub\wwwroot\sisaac\profile\imgUsers\" + dt.Rows[0][0].ToString();
        }
```

```
            else
                mensaje = null;
            return mensaje;
        }

        [WebMethod(Description = " Metodo que actualiza la imagen de un Usuario")]
        public void Actualizar_imagen(string id, string new_imagen) {
            string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
            SqlConnection myConnection = new SqlConnection(strConnectionString);
            SqlCommand cmd = new SqlCommand();
            cmd.CommandType = CommandType.Text;
            cmd.CommandText = "UPDATE usuarios SET imagen = '" + new_imagen + "'" + "WHERE
IdUsuario = '" + id + "'";
            cmd.Connection = myConnection;
            myConnection.Open();
            cmd.ExecuteNonQuery();
            myConnection.Close();
        }
    }
}
```

**Entorno principal.**

Es nombrado entorno principal porque a partir de él se tiene acceso a todo SISAAC. Este entorno invoca los métodos de los Servicios Web Grupos.asmx y Sesion.asmx que son descritos a continuación.

El Servicio Web Grupos.asmx contiene los métodos: ConsultaGrupos, CrearGrupo, ConsultaexisteGrupo, Retorna_Idgrupo, Retorna_nombregrupo, Propietariogrupo, para la creación y acceso de los grupos.

Grupos.asmx

```
using System.Web.Services;
using System.Configuration;
using System.Data.SqlClient;
using System.Data;
using System;

[WebService(Namespace = "http://tempuri.org/")]
public class Grupos : System.Web.Services.WebService{
    protected SqlDataAdapter da;
    protected DataSet ds;
    private DataRow dr;
    protected SqlConnection myConnection = new SqlConnection();

    public Grupos () {  }

    [WebMethod(Description = " Metodo que retorna un datatable con todos los grupos")]
    public DataTable ConsultaGrupos()   {
        string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
        SqlConnection myConnection = new SqlConnection(strConnectionString);
```

```
        string strCommandText = "SELECT idgrupo,nomgrupo FROM Grupo ";
        da = new SqlDataAdapter(strCommandText, myConnection);
        SqlCommandBuilder cb = new SqlCommandBuilder(da);
        da.UpdateCommand = cb.GetUpdateCommand();
        da.InsertCommand = cb.GetInsertCommand();
        da.DeleteCommand = cb.GetDeleteCommand();
        da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
        DataTable dt= new DataTable();
        try{  da.Fill(dt); }
        catch (Exception ex) { Console.WriteLine(ex.Message); }
        finally   {
            if (myConnection != null)
                myConnection.Close();
        }
        return dt;
    }

    [WebMethod(Description = " Metodo que checa si existe un grupo determinado")]
    public bool ConsultaExisteGrupo(string grupo)    {
        string                                          strConnectionString                                          =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
        SqlConnection myConnection = new SqlConnection(strConnectionString);
        string strCommandText = "SELECT * FROM Grupo  WHERE nomgrupo= '" + grupo + "'";
        da = new SqlDataAdapter(strCommandText, myConnection);
        ds = new DataSet();
        try{da.Fill(ds, "Grupo");}
        catch (Exception ex) { Console.WriteLine(ex.Message); }
        finally{
            if (myConnection != null)
                myConnection.Close();
        }
        if (ds.Tables["Grupo"].Rows.Count > 0)
            return true;
        else
            return false;
    }

    [WebMethod(Description = " Metodo que Crea un grupo")]
    public string CrearGrupo(string nomgrupo, string descripcion, string nucgrupo,string fecha) {
        string                                          strConnectionString                                          =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
        SqlConnection myConnection = new SqlConnection(strConnectionString);
        string strCommandText = "SELECT * FROM Grupo ";
        da = new SqlDataAdapter(strCommandText, myConnection);
        SqlCommandBuilder cb = new SqlCommandBuilder(da);
        da.UpdateCommand = cb.GetUpdateCommand();
        da.InsertCommand = cb.GetInsertCommand();
        da.DeleteCommand = cb.GetDeleteCommand();
        da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
        DataTable dt = new DataTable();
        try{
            da.Fill(dt);
            dr = dt.NewRow();
            dr["nomgrupo"] = nomgrupo;
            dr["descripcion"] = descripcion;
            dr["nucgrupo"] = nucgrupo;
```

```
            dr["fechacreacion"] = fecha;
            dt.Rows.Add(dr);
            da.Update(dt);
            dt.AcceptChanges();
        }
        catch (Exception ex) {Console.WriteLine(ex.Message); }
        finally{
            if (myConnection != null)
                myConnection.Close();
        }
        return "Grupo creado exitosamente...!!!";
    }

    [WebMethod(Description = " Metodo que retorna el Id de un grupo a partir del nombre del grupo")]
    public string Retorna_Idgrupo(string nomgrupo)    {
        string strConnectionString = ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
        SqlConnection myConnection = new SqlConnection(strConnectionString);
       string strCommandText = "SELECT idgrupo FROM Grupo where nomgrupo='"+nomgrupo + "'";
        da = new SqlDataAdapter(strCommandText, myConnection);
        SqlCommandBuilder cb = new SqlCommandBuilder(da);
        da.UpdateCommand = cb.GetUpdateCommand();
        da.InsertCommand = cb.GetInsertCommand();
        da.DeleteCommand = cb.GetDeleteCommand();
        da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
        DataTable dt = new DataTable();
        string id;
        try  {da.Fill(dt); }
        catch (Exception ex) {Console.WriteLine(ex.Message); }
        finally{
            if (myConnection != null)
                myConnection.Close();
        }
        if (dt.Rows.Count > 0)
            id = dt.Rows[0][0].ToString();
        else
            id = "false";
        return id;
    }

    [WebMethod(Description = " Metodo que retorna el nombre de un grupo a partir del ID del grupo")]
    public string Retorna_nombregrupo(string idgrupo)  {
        string strConnectionString = ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
        SqlConnection myConnection = new SqlConnection(strConnectionString);
        string strCommandText = "SELECT nomgrupo FROM Grupo where idgrupo='" + idgrupo + "'";
        da = new SqlDataAdapter(strCommandText, myConnection);
        SqlCommandBuilder cb = new SqlCommandBuilder(da);
        da.InsertCommand = cb.GetInsertCommand();
        da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
        DataTable dt = new DataTable();
        string id;
        try{da.Fill(dt);   }
        catch (Exception ex) {Console.WriteLine(ex.Message); }
```

```
        finally{
            if (myConnection != null)
                myConnection.Close();
        }
        if (dt.Rows.Count > 0)
            id = dt.Rows[0][0].ToString();
        else
            id = "false";
        return id;
    }

    [WebMethod(Description = " Metodo que checa si un usuario es el propietario de un grupo")]
    public bool PropietarioGrupo(string grupo, string usuario) {
        string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
        SqlConnection myConnection = new SqlConnection(strConnectionString);
        string strCommandText = "SELECT * FROM Grupo  WHERE nomgrupo = '" + grupo + "' AND
nucgrupo = '"+usuario+"'";
        da = new SqlDataAdapter(strCommandText, myConnection);
        ds = new DataSet();
        try{da.Fill(ds, "Grupo");}
        catch (Exception ex) {Console.WriteLine(ex.Message); }
        finally{
            if (myConnection != null)
                myConnection.Close();
        }
        if (ds.Tables["Grupo"].Rows.Count > 0)
            return true;
        else
            return false;
    }
}
```

El Servicio Web Sesion.asmx, con sus métodos: CrearSesion, EliminaSesion y ConsultaExisteSesion, se encarga de verificar si un usuario ya tiene una sesión en un grupo, en caso contrario la crea o la elimina de ser necesario.

Sesion.asmx

```
using System.Web.Services;
using System.Configuration;
using System.Data.SqlClient;
using System.Data;
using System;

namespace Samples.Aspnet{
    [WebService(Namespace = "http://tempuri.org/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.Web.Script.Services.ScriptService]
    public class Sesion : System.Web.Services.WebService{
        DataRow dr;
        SqlDataAdapter da;
        DataSet ds;
        protected SqlConnection myConnection = new SqlConnection();

        public Sesion() { }
```

```
    [WebMethod(Description = " Metodo que Crea una sesion de Usuario en un Grupo")]
    public void CrearSesion(string IdUsuario, string IdGrupo, string fecha){
        string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
        SqlConnection myConnection = new SqlConnection(strConnectionString);
        string strCommandText = "SELECT * FROM Sesion";
        da = new SqlDataAdapter(strCommandText, myConnection);
        SqlCommandBuilder cb = new SqlCommandBuilder(da);
        da.UpdateCommand = cb.GetUpdateCommand();
        da.InsertCommand = cb.GetInsertCommand();
        da.DeleteCommand = cb.GetDeleteCommand();
        da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
        DataTable dt = new DataTable();
        try{
            da.Fill(dt);
            dr = dt.NewRow();
            dr["IdUsuario"] = IdUsuario;
            dr["IdGrupo"] = IdGrupo;
            dr["fecha"] = fecha;
            dt.Rows.Add(dr);
            da.Update(dt);
            dt.AcceptChanges();
        }
        catch (Exception ex) {Console.WriteLine(ex.Message); }
        finally{
            if (myConnection != null)
                myConnection.Close();
        }
    }

    [WebMethod(Description = " Metodo que checa si existe una sesion del usuario en un grupo
determinado")]
    public bool ConsultaExisteSesion(string IdUsuario, string IdGrupo) {
        string strConnectionString =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
        SqlConnection myConnection = new SqlConnection(strConnectionString);
        string strCommandText = "SELECT idsesion FROM Sesion WHERE IdUsuario='" +
IdUsuario + "' and IdGrupo='" + IdGrupo + "'";
        da = new SqlDataAdapter(strCommandText, myConnection);
        ds = new DataSet();
        try{ da.Fill(ds, "Sesion");}
        catch (Exception ex) {Console.WriteLine(ex.Message); }
        finally{
            if (myConnection != null)
                myConnection.Close();
        }
        if (ds.Tables["Sesion"].Rows.Count > 0)
            return true;
        else
            return false;
    }

    [WebMethod(Description = " Metodo Elimina la sesion de un Usuario en un grupo
determinado")]
    public void EliminaSesion(string idusuario, string idgrupo) {
```

```
        string strConnectionString = ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
        SqlConnection myConnection = new SqlConnection(strConnectionString);
        SqlCommand cmd = new SqlCommand();
        cmd.CommandType = CommandType.Text;
        cmd.CommandText = "DELETE FROM Sesion WHERE idusuario = '" + idusuario + "' AND idgrupo='" + idgrupo + "'";
        cmd.Connection = myConnection;
        myConnection.Open();
        cmd.ExecuteNonQuery();
        myConnection.Close();
    }
  }
}
```

**Grupos.**

Ya que la creación de grupos se hace en la página principal, la página web Grupos.aspx es el espacio en donde se pueden crear foros y acceder al espacio para compartir archivos sin embargo esos dos temas no son abordados en esta tesis. No obstante, si hace uso de un Servicio Web llamado MensajesGrupo.asmx por medio del cual se almacenan mensajes del grupo en la base de datos. Este Servicio Web muestra a continuación.

MensajeGrupo.asmx

```
using System.Web.Services;
using System.Configuration;
using System.Data.SqlClient;
using System.Data;
using System;

namespace Samples.Aspnet{
   [WebService(Namespace = "http://tempuri.org/")]
   [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
   [System.Web.Script.Services.ScriptService]
   public class MensajesGrupo : System.Web.Services.WebService{
      protected SqlDataAdapter da;
      protected DataSet ds;
      private DataRow dr;
      protected SqlConnection myConnection = new SqlConnection();

  [WebMethod(Description = " Metodo que retorna todos los mensajes pertenecientes a un grupo")]
      public string[] ConsultaMensajes(string idgrupo) {
         string strConnectionString = ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
         SqlConnection myConnection = new SqlConnection(strConnectionString);
         string strCommandText = "SELECT nucmensaje,mensaje,fechacreacion FROM MensajesGrupo where idgrupo='" + idgrupo + "' ORDER BY fechacreacion";
         da = new SqlDataAdapter(strCommandText, myConnection);
         SqlCommandBuilder cb = new SqlCommandBuilder(da);
         da.InsertCommand = cb.GetInsertCommand();
         da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
         DataTable dt = new DataTable();
```

```
            string[] mensaje;
            try{ da.Fill(dt); }
            catch (Exception ex) {Console.WriteLine(ex.Message); }
            finally{
                if (myConnection != null)
                    myConnection.Close();
            }
            mensaje = new string[dt.Rows.Count*3];
            if (dt.Rows.Count > 0){
                for (int i = 0; i < dt.Rows.Count; i++){
                    mensaje[i*3] = dt.Rows[i][0].ToString();
                    mensaje[i *3+ 1] = dt.Rows[i][1].ToString();
                    mensaje[i *3+ 2] = dt.Rows[i][2].ToString();
                }
            }
            return mensaje;
        }

    [WebMethod(Description = " Metodo que checa si existen mensajes en un grupo determinado")]
      public bool ConsultaExistenMensajes(string idgrupo) {
            string                                          strConnectionString                                          =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
            SqlConnection myConnection = new SqlConnection(strConnectionString);
            string strCommandText = "SELECT * FROM MensajesGrupo  WHERE idgrupo= '" + idgrupo
+ "' ORDER BY fechacreacion";
            da = new SqlDataAdapter(strCommandText, myConnection);
            ds = new DataSet();
            try{  da.Fill(ds, "MensajesGrupo");}
            catch (Exception ex)   {Console.WriteLine(ex.Message);  }
            finally{
                if (myConnection != null)
                    myConnection.Close();
            }
            if (ds.Tables["MensajesGrupo"].Rows.Count > 0)
                return true;
            else
                return false;
        }

        [WebMethod(Description = " Metodo que Crea un Mensaje")]
        public void AddMessage(string idgrupo,string mensaje,string usuario,string fecha)  {
            string                                          strConnectionString                                          =
ConfigurationManager.ConnectionStrings["SqlConnectionString"].ConnectionString;
            SqlConnection myConnection = new SqlConnection(strConnectionString);
            string strCommandText = "SELECT * FROM mensajesgrupo " ;
            da = new SqlDataAdapter(strCommandText, myConnection);
            SqlCommandBuilder cb = new SqlCommandBuilder(da);
            da.UpdateCommand = cb.GetUpdateCommand();
            da.InsertCommand = cb.GetInsertCommand();
            da.DeleteCommand = cb.GetDeleteCommand();
            da.MissingSchemaAction = MissingSchemaAction.AddWithKey;
            DataTable dt = new DataTable();
            try{
                da.Fill(dt);
                dr = dt.NewRow();
                dr["idgrupo"] = idgrupo;
```

```
            dr["mensaje"] = mensaje;
            dr["nucmensaje"] = usuario;
            dr["fechacreacion"] = fecha;
            dt.Rows.Add(dr);
            da.Update(dt);
            dt.AcceptChanges();
        }
        catch (Exception ex) {Console.WriteLine(ex.Message); }
        finally{
            if (myConnection != null)
                myConnection.Close();
        }
    }
  }
}
```

**Panel de control.**

El panel de control es una GUI (interfaz grafica de usuario) en la cual se administra el perfil y preferencias de un usuario, por consiguiente por medio de esta interfaz se accede a otras páginas web que invocan a todos los Servicios Web ya mencionados e incluso a otros que no se trataran en esta tesis. A continuación se muestra el código de la pagina web usercp.aspx correspondiente al panel de control.

Usercp.aspx

```
<%@Page Language="C#"AutoEventWireup="true"CodeFile="usercp.aspx.cs"Inherits="Private"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title>SisAAC -Panel de Control-</title>
  <link href="App_Themes/usercp.css" rel="stylesheet" type="text/css" />
</head>
<body >
<form id="form1" runat="server">
   <div id="encabezado">
      <div id="logo">
         <div id="login">Usuario:
            <asp:Label ID="lbl_nusuario" runat="server" ForeColor="#E1BA71"></asp:Label>
         </div>
            <asp:ScriptManager ID="ScriptManager1" runat="server">
               <Services>
                                    <asp:ServiceReference Path="http://localhost/BlogWS/Blog.asmx" />
                           </Services>
                     <Scripts>
                        <asp:ScriptReference Path="~/App_Code_JS/BlogAndTools.js" />
                        <asp:ScriptReference Path="~/App_Code_JS/Utilerias.js" />
                     </Scripts>
         </asp:ScriptManager>
      </div>
      <div id="menu">
      </div>
   </div>
```

```
    <div id="principal"><br />
        <div id="barraTop">
        </div>
                    <div class="display">
                        <h2 class="tither"><b>Bienvenido a la comunidad SisAAC</b></h2>
                    </div>
        <div id="lateral">
    </div>
    <div id="footer">
    Copyrigth © 2010 . All Rights Reserved. SISAAC ver 1.0<br />
        Benemerita Universidad Autonoma de Puebla<br />
        Facultad de Ciencias de la Computación
        <asp:Label ID="lbl_idusuario" runat="server"></asp:Label>
    </div>
            </div>
    </form>
</body>
</html>
```

A continuación se presentan el scripts de JavaScript MensajesGrupo.js el cual es utilizado para mostrar los mensajes del grupo en este script se hace uso de DOM para la maquetación.

MensajesGrupo.js

```
function mostrar_mensajes() {
    var id_grupo = document.getElementById('lbl_idgrupo').innerHTML;
    Samples.Aspnet.MensajesGrupo.ConsultaMensajes(id_grupo,success_mensajes);
}

function success_mensajes(resultado) {
    document.getElementById("mensajesGrupo").innerHTML = "";
    for (var i = 0; i < (resultado.length / 3); i++) {
        var tabla = '<div class=Respuestas><table><TBODY>';
        tabla  +=  '<tr><td    style="width:695px"  valign="bottom"  text-align="left"  class="from"><B>De:
</B><I>'+resultado[i*3]+'</I></td></tr>';
        tabla +='<tr><td  valign="bottom" text-align="left" class="from"><B>Fecha: </B>' + resultado[i *
3 + 2] + '</td></tr>';
        tabla += '<tr><td valign="bottom" text-align="left" class="from"><B>Mensaje: </B></td></tr>';
        tabla  +='<tr><td  valign="bottom"  text-align="justify"  class="from">' + resultado[i * 3 + 1] +
'</td></tr>';
        tabla += '</TBODY></table></div>';
        document.getElementById("mensajesGrupo").innerHTML += tabla + '<br/>';
    }
}

function agrega_comentario() {
    var m = document.getElementById('text_coment').value;
    var id_grupo = document.getElementById('lbl_idgrupo').innerHTML;
    var usuario = document.getElementById('lbl_nusuario').innerHTML;
    Samples.Aspnet.MensajesGrupo.AddMessage(id_grupo, m, usuario, fecha());
    alert("Tu comentario ha sido agregado, Gracias...!!");
    document.getElementById('text_coment').innerHTML = "";
    mostrar_mensajes();
}
```

```
function fecha() {
    var f = new Date();
    var d = f.getDate();
    if (d < 10)
        d = "0" + d + "/";
    else
        d = d + "/";
    var m = f.getMonth();
    if (m < 10)
        m = "0" + m + "/";
    else
        m = m + "/";
    var y = f.getFullYear();
    return d + m + y;
}
```

A continuación se presenta el script de CSS perfil.css el cual es utilizado para la creación de la interfaz del entorno principal.

```
body {
margin-top: 0px;
padding: 0;
background:#dfdfc5;
color:#000000;
}
#encabezado {
width: 1075px;
height:200px;
margin: 0 auto;
background:#960f00;
}
#login{
width:300px;
height:20px;
color:white;
font: italic 13px Georgia, "Times New Roman", Times, serif;
text-align:left;
float:right;
padding-top : 5px;
}
#contenedor
{
clear:both;
width: 1075px;
height:auto;
margin: 0 auto;
color:#000000;
background-color:#ffffff;
}
#logo {

padding : 5px 0px 0px 0px;
}
#cuerpo{
width:1075;
height:150px;
background:url(../images/logo_sisaac.png);
} //end logo
#cabecera {
height: 28px;
margin: 0 auto;
background: #960f00;
}
#menu{
clear:both;
width:1075px;
height:25px;
background:url(../images/barra_menus.png);
text-align:center;
padding-top:5px; Times, serif;
text-transform: uppercase;
}
A.enlacenav, A.enlacenav:VISITED,
A.enlacenav:ACTIVE, A.enlacenav:FOCUS,
A.enlacenav:LINK{
color: #960f00;
text-decoration:none;
}
A.enlacenav:HOVER
{
background:#960f00;
color: #fe9f0b;
}
#usuario{
float: right;
width: 195px;

width:800px;
margin-left: 0px;
padding: 12px 0px 10px 0px;
```

```
background-color :#ffffff;
float:left;
}
.herramientas{
width:auto;
height:auto;
border: 1px solid #cccccc;
color:#000000;
font-size:13px;
}
#lateral{
background-color: #ffffff;
float:right;
width:200px;
height:auto;
margin-right:25px;
font-size:13px;
}
.titlat{
background-color:#f4f8f0;
color:#000000;
font-size:13px;
padding: 7px 3px 7px 8px;
font-weight : normal;
letter-spacing : 2px;
margin: 0px 0px 9px 0px;
}
.cuerpolateral{
padding: 5px 4px 13px 10px;
background-color: #ffffff;
}
#botoncambiar {
padding-top : 5px;
padding-left: 106px;
}
#otras ul{
margin : 5px 10px 0px 0px;
padding: 0px 0px 0px 4px;
list-style: none;
}
#otras li{
padding-left: 14px;
margin-bottom: 10px;
}
#pie {
clear:both;
width:1075px;
margin:0 auto;
height:100px;
text-align : center;
background:#dfdfc5;
color:Olive;
padding-top:10px;
font-family:Times New Roman;
font-size: 10px;
} // end pie
#borde{
border: 2px solid #cccccc;
text-align: left;
width: 900px;
margin: auto;
}
#imagenperfil{
float:left;
width:100px;
text-align:right;
}
#datosusuario{
float:right;
width:600px;
text-align:center;
color:#000000;
}
#Mensaje_principal{
float:none;
width:auto;
text-align:center;
color:#000000;
}
#tabla{
width:auto;
font: italic 15px Georgia, "Times New
Roman", Times, serif;
text-align:left;
color:#000000;
}
#com_grupo
{
width:600px;
float:none;
height:auto;
padding-left:100px; Roman", Times, serif;
}
#mensajesGrupo
{
float:none;
width:600px;
height:auto;
padding-left:100px;
Roman", Times, serif;
color:white;
}
#mensajesGrupo p{
color : #ffffff; font: italic 15px Georgia, "Times
NewRoman", Times, serif;
```

```
oTable{
float:inherit;
width:697px;
text-align:left;
font:    15px  Georgia,  "Times  New  Roman",
Times, serif;
color:#ffffff;
}
#oTBody1
{
color:#000000;
}
.subirA{
width:800px;
padding: 12px 0px 10px 0px;
background-color :#f4f8f0;
float:left;
}
```

Finalmente se muestra el código del archivo **sisaac.sql** el cual corresponde a la creación de todas las tablas de la base de datos de SISAAC. Dicho código se genera a partir de Erwin Data Modeler en su versión 7.0.

```
CREATE TABLE Blog(
        Idblog          int IDENTITY (1000,1) ,
        IdUsuario       char(9)  NOT NULL
)go
ALTER TABLE Blog
        ADD CONSTRAINT XPKBlog PRIMARY KEY  CLUSTERED (Idblog ASC)
go
CREATE TABLE Contactos(
        iduscontactado      char(9)  NULL ,
        IdUsuario       char(9)  NOT NULL ,
        clave           int IDENTITY (1000,1)
)go
ALTER TABLE Contactos
        ADD CONSTRAINT XPKContactos PRIMARY KEY  CLUSTERED (clave ASC)
go
CREATE TABLE Foro(
        nomforo         varchar(255)  NULL ,
        nucforo         varchar(25)  NULL ,
        descripcion     text  NULL ,
        fechacreacion       varchar(10)  NULL ,
        Idforo          int IDENTITY (1000,1) ,
        IdGrupo         int  NOT NULL
)go
ALTER TABLE Foro
        ADD CONSTRAINT XPKForo PRIMARY KEY  CLUSTERED (Idforo ASC)
go
CREATE TABLE Grupo(
        IdGrupo         int IDENTITY (1000,1) ,
        nomgrupo        char(255)  NOT NULL ,
        descripcion     text  NULL ,
        nucgrupo        varchar(25)  NULL ,
        fechacreacion       char(10)  NULL
)go
ALTER TABLE Grupo
        ADD CONSTRAINT XPKGrupo PRIMARY KEY  CLUSTERED (IdGrupo ASC)
go
CREATE TABLE ImagenesBlog(
        idImagen        int IDENTITY (1000,1) ,
        MimeType        varchar(255)  NULL ,
        nombre          varchar(255)  NULL ,
```

```
        comentario        text  NULL ,
        fecha           datetime  NULL ,
        Idblog          int  NOT NULL ,
        ruta           varchar(255)  NULL
)go
ALTER TABLE ImagenesBlog
        ADD CONSTRAINT XPKImagenes PRIMARY KEY  CLUSTERED (idImagen ASC)
go
CREATE TABLE MBInvitado(
        Idmensaje        int IDENTITY (1000,1) ,
        mensaje          text  NULL ,
        fecha           datetime  NULL ,
        Idblog          int  NOT NULL ,
        nusuario         varchar(25)  NULL
)go
ALTER TABLE MBInvitado
        ADD CONSTRAINT XPKMBInvitado PRIMARY KEY  CLUSTERED (Idmensaje ASC)
go
CREATE TABLE MensajesBlog(
        Idmensaje        int IDENTITY (1000,1) ,
        mensaje          text  NULL ,
        fecha           datetime  NULL ,
        Idblog          int  NOT NULL
)go
ALTER TABLE MensajesBlog
        ADD CONSTRAINT XPKMensajesBlog PRIMARY KEY  CLUSTERED (Idmensaje ASC)
go
CREATE TABLE MensajesGrupo(
        IdMensaje        int IDENTITY (1000,1) ,
        mensaje          text  NULL ,
        fechacreacion      char(10)  NULL ,
        nucmensaje         varchar(25)  NULL ,
        IdGrupo          int  NOT NULL
)go
ALTER TABLE MensajesGrupo
        ADD CONSTRAINT XPKMensajesGrupo PRIMARY KEY  CLUSTERED (IdMensaje ASC)
go
CREATE TABLE MensajesUsuarios(
        clave           int IDENTITY (1000,1) ,
        idusdestino       char(9)  NOT NULL ,
        mensaje          text  NULL ,
        fecha           char(10)  NULL ,
        asunto           text  NULL ,
        IdUsuario        char(9)  NOT NULL
)go
ALTER TABLE MensajesUsuarios
        ADD CONSTRAINT XPKMensajesUsuarios PRIMARY KEY  CLUSTERED (clave ASC)
go
CREATE TABLE Preguntas(
        Idpregunta        int IDENTITY (1000,1) ,
        nupregunta         varchar(25)  NULL ,
        pregunta         text  NULL ,
        fecha           char(10)  NULL ,
        hora            char(8)  NULL ,
        Idforo          int  NOT NULL
)go
```

```
ALTER TABLE Preguntas
	ADD CONSTRAINT XPKPreguntas PRIMARY KEY  CLUSTERED (Idpregunta ASC)
go
CREATE TABLE Respuestas(
	Idrespuesta	 int IDENTITY (1000,1) ,
	nurespuesta	  varchar(25)  NULL ,
	respuesta	 text  NULL ,
	fecha	 char(10)  NULL ,
	hora	 char(8)  NULL ,
	Idpregunta	  int  NOT NULL
)go
ALTER TABLE Respuestas
	ADD CONSTRAINT XPKRespuestas PRIMARY KEY  CLUSTERED (Idrespuesta ASC)
go
CREATE TABLE Sesion(
	IdSesion	 int IDENTITY (1000,1) NOT FOR REPLICATION ,
	IdGrupo	 int  NOT NULL ,
	IdUsuario	 char(9)  NOT NULL ,
	fecha	 char(10)  NULL
)go
ALTER TABLE Sesion
	ADD CONSTRAINT XPKSesion PRIMARY KEY  CLUSTERED (IdSesion ASC,IdUsuario
ASC,IdGrupo ASC)go
CREATE TABLE UploadFiles(
	clave	 int IDENTITY (1000,1) ,
	nombre	  varchar(255)  NULL ,
	ruta	 varchar(255)  NULL ,
	Mimetype	  varchar(255)  NULL ,
	comentario	  text  NULL ,
	fecha	 datetime  NULL ,
	usuario	 varchar(25)  NULL ,
	IdGrupo	  int  NOT NULL
)go
ALTER TABLE UploadFiles
	ADD CONSTRAINT XPKUploadFiles PRIMARY KEY  CLUSTERED (clave ASC)
go
CREATE TABLE Usuarios(
	IdUsuario	 char(9)  NOT NULL ,
	pswd	 varchar(40)  NOT NULL ,
	nombre	  varchar(25)  NOT NULL ,
	appat	 varchar(25)  NOT NULL ,
	apmat	 varchar(25)  NOT NULL ,
	facultad	 varchar(50)  NULL ,
	carrera	 varchar(50)  NULL ,
	campus	  varchar(25)  NULL ,
	email	 varchar(35)  NULL ,
	fechan	 char(10)  NULL ,
	nusuario	  varchar(25)  NOT NULL ,
	comentario	  text  NULL ,
	genero	 varchar(6)  NULL ,
	imagen	 varchar(255)  NULL ,
	estado	 varchar(5)  NULL
)go
ALTER TABLE Usuarios
	ADD CONSTRAINT XPKUsuarios PRIMARY KEY  CLUSTERED (IdUsuario ASC)
go
```

```
ALTER TABLE Blog
	ADD CONSTRAINT R_102 FOREIGN KEY (IdUsuario) REFERENCES Usuarios(IdUsuario)
		ON DELETE CASCADE
		ON UPDATE CASCADE
go
ALTER TABLE Contactos
	ADD CONSTRAINT R_100 FOREIGN KEY (IdUsuario) REFERENCES Usuarios(IdUsuario)
		ON DELETE CASCADE
		ON UPDATE CASCADE
go
ALTER TABLE Foro
	ADD CONSTRAINT R_93 FOREIGN KEY (IdGrupo) REFERENCES Grupo(IdGrupo)
		ON DELETE NO ACTION
		ON UPDATE NO ACTION
go
ALTER TABLE ImagenesBlog
	ADD CONSTRAINT R_103 FOREIGN KEY (Idblog) REFERENCES Blog(Idblog)
		ON DELETE CASCADE
		ON UPDATE CASCADE
go
ALTER TABLE MBInvitado
	ADD CONSTRAINT R_106 FOREIGN KEY (Idblog) REFERENCES Blog(Idblog)
		ON DELETE CASCADE
		ON UPDATE CASCADE
go
ALTER TABLE MensajesBlog
	ADD CONSTRAINT R_104 FOREIGN KEY (Idblog) REFERENCES Blog(Idblog)
		ON DELETE CASCADE
		ON UPDATE CASCADE
go
ALTER TABLE MensajesGrupo
	ADD CONSTRAINT R_96 FOREIGN KEY (IdGrupo) REFERENCES Grupo(IdGrupo)
		ON DELETE NO ACTION
		ON UPDATE NO ACTION
go
ALTER TABLE MensajesUsuarios
	ADD CONSTRAINT R_101 FOREIGN KEY (IdUsuario) REFERENCES Usuarios(IdUsuario)
		ON DELETE CASCADE
		ON UPDATE CASCADE
go
ALTER TABLE Preguntas
	ADD CONSTRAINT R_98 FOREIGN KEY (Idforo) REFERENCES Foro(Idforo)
		ON DELETE NO ACTION
		ON UPDATE NO ACTION
go
ALTER TABLE Respuestas
	ADD CONSTRAINT R_97 FOREIGN KEY (Idpregunta) REFERENCES
Preguntas(Idpregunta)
		ON DELETE NO ACTION
		ON UPDATE NO ACTION
go
ALTER TABLE Sesion
	ADD CONSTRAINT R_71 FOREIGN KEY (IdGrupo) REFERENCES Grupo(IdGrupo)
		ON DELETE NO ACTION
		ON UPDATE NO ACTION
go
```

```
ALTER TABLE Sesion
	ADD CONSTRAINT R_73 FOREIGN KEY (IdUsuario) REFERENCES Usuarios(IdUsuario)
		ON DELETE CASCADE
		ON UPDATE CASCADE
go
ALTER TABLE UploadFiles
	ADD CONSTRAINT R_107 FOREIGN KEY (IdGrupo) REFERENCES Grupo(IdGrupo)
		ON DELETE NO ACTION
		ON UPDATE NO ACTION
go
```

**REFERENCIAS.**


[Avda, 04] Avda Reina Mercedes, Bases de Datos Relacionales de Codd Estructuras y restricciones. Departamento de Lenguajes y Sistemas Informáticos, universidad de Sevilla, Marzo 2004.

[Blackboad, 05 ] Manual del profesor, junio 2005, http://www.blackboard.com/

[Cashman, 84] P. Cashman and Sluizer, S, 'XCP: an experimental tool for supporting officeprocedures'. In R.W. Taylor(ed.), Proceedings of the IEEE first international conference on office automation, NewOrleans, LA, USA, 1984. IEE Computer Society Press. Silver Spring, MD, USA, 1994, p.73-80.

[Christensen, 01] E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana. (2001) "Web Services Description Language (WSDL) 1.1 http://www.w3.org/TR/wsdl.

[Dokeos, 06] Creando y publicando cursos virtuales con DOKEOS 1.8, abril 2007, http://www.enko.com.mx/dokeos/ .

[Eguíluz, 09] Eguíluz Pérez Javier, Introducción a CSS, Mayo 2009

[Eguíluz, 09] Eguíluz Pérez Javier, Introducción a JavaScript, Marzo 2009

[Ellis, 1991] Ellis, CA.; Guibs, S.J.; Rein, G.L: "Groupware: Some Issues and Experiences", Communication of the ACM Vol. 34, No 1(January 1991), pp 38-58.

[Friss, 05] Friss de Kereki Guerrero, Inés, 'Tesis Doctoral: Modelo para la creación de entornos de Aprendizaje basado en Técnicas de Gestión del Conocimiento', 2005

[Johnson-Lenz, 81] Peter Johnson-Lenz, Trudy Johnson-Lenz. "Consider the Groupware: Design and Group Process Impacts on Communication in the Electronic Medium," in Hiltz, S. and Kerr, E., Studies of Computer-Mediated Communications System: A Synthesis of the finding, research Report #16, computerized Conferencing and Communications Center, New Jersey Institute of Technology, Newark,

[Martínez 05] Martínez Carreras Ma. Antonia, 'Aplicación a la Plataforma de Aprendizaje '. Departamento de la ingeniería de la Información y las comunicaciones. Universidad de Murcia. 2005.

[Microsoft, ] Microsoft "Formatos de conexión de servicios Web XML (HTTP, SOAP)": http://msdn.microsoft.com/es-es/library/a1tx28sw(VS.80).aspx

[MySQL, 03] MySQL hispano , Normalización de Bases de Datos , May 2003.

[Moodle,2007] Acerca de Moodle, agosto 2007, http://docs.moodle.org/es/Acerca_de_Moodle.

[OASIS, 02] OASIS. (2002) “UDDI Version 2.04 API Specification”. http://uddi.org/pubs/ProgrammersAPI v2.htm.

[OASIS, 02] OASIS. (2002) “UDDI Version 2.04 API Specification”. http://uddi.org/pubs/ProgrammersAPI v2.htm.

[Pressman, 02] Roger S. Pressman & Associates, Inc. Ingenieria Del Software un Enfoque Práctico, Quinta edición, McGRAW-HILLDNTERAMERICANA De España, *S.* A. U., 2002

[Rumbaugh, 00] J. Rumbbaugh, I.J acobson, & G. Booch, El Lenguaje Unificado de Modelado Manual de Referencias, Pearson Educación, S.A., Madrid, 2000.

[Sarin, 85] Sarin, S. and I. Greif, ‘Computer-based real time conferencing systems’. Computer, 18 (October 1985), 10, p.33-45.

[Silberschatz, *02*] Silberschatz Abraham, Henry F. Korth y S. Sudarshan, Fundamentos De Bases de Datos.Cuarta edición, McGRAW-HILL/INTERAMERICANA DE ESPAÑA, *Bell Laboratories. Sudarshan S. &Instituto Indio de Tecnologia, Bombay, 2002.*

[Sommerville, 05] Sommerville Ian, Ingenieria del software, séptima edición, Addisson –Wesley, Madrid, 2005

[Warmer, 99] Jos B. Warmer, Anneke G. Kleppe. The Object Restriction Language: Precise Modeling whith UML, Addison-Wesley, Reading, Mass., 1990.

[W3C, 03] W3C. (2003) “SOAP Version 1.2 Part 1: Messaging Framework”. ttp://www.w3.org/TR/soap12-part1/