



**UNIVERSIDAD NACIONAL AUTÓNOMA
DE MÉXICO**

**FACULTAD DE ESTUDIOS SUPERIORES
ARAGÓN**

**Análisis y Desarrollo del Sistema de Asignaciones Docentes
para la FES Cuautitlán**

TRABAJO ESCRITO EN LA MODALIDAD DE:

TESIS

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

PRESENTA:

LUIS ENRIQUE BALTAZAR PAZ

ASESOR:

DR. ROBERTO MURCIO VILLANUEVA



SAN JUAN DE ARAGÓN, ESTADO DE MÉXICO

2012



Universidad Nacional
Autónoma de México

Dirección General de Bibliotecas de la UNAM

Biblioteca Central



UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

ÍNDICE

INTRODUCCIÓN	1
CAPÍTULO 1 ANTECEDENTES	2
1.1. Misión y Visión de la FESC	2
1.1.1.Misión	2
1.1.2.Visión.....	2
1.2. Contexto actual de la FESC	2
1.3. Organización institucional de la FESC	3
1.4. Análisis de la problemática	8
1.5. Propuesta de solución	11
1.6. Objetivos	12
1.6.1.Objetivos generales	12
1.6.2.Objetivos específicos	12
1.7. Metas	13
CAPÍTULO 2 SISTEMAS DE INFORMACIÓN Y SISTEMAS DISTRIBUIDOS ...	14
2.1. ¿Qué es un sistema de información?	14
2.2. Ciclo de vida de software	14
2.3. Modelos de ciclo de vida	17
2.3.1.Modelo en cascada	17
2.3.2.Modelo de desarrollo evolutivo.....	18
2.3.3.Modelo de desarrollo basado en componentes	18
2.3.4.Modelo de desarrollo en espiral	19
2.4. Sistemas distribuidos	20
2.4.1.Definición de sistemas distribuidos.....	20
2.4.2.Modelo cliente-servidor	21
2.4.3.Arquitectura de dos capas	22
2.4.4.Arquitectura de tres capas	23
2.4.5.Arquitectura multi-capas	24
2.5.Herramientas CASE	25
CAPÍTULO 3 SISTEMAS GESTORES DE BASES DE DATOS (SGBD)	26
3.1.Bases de datos	26
3.1.1.Arquitectura de las Bases de Datos	26
3.1.2.Características de las bases de datos	27
3.1.3.Modelo de datos	28
3.1.4.Modelo relacional	28
3.1.5.Bases de datos relacionales	29
3.1.6.Reglas de Codd	29
3.1.7.Lenguaje de consulta relacional	31
3.1.8.Modelo Entidad - Relación	31
3.1.9.Normalización.....	32
3.2.Sistemas Gestores de Bases de Datos	33
3.2.1.Clasificación de los Sistemas Gestores de Bases de Datos	34

CAPÍTULO 4 ANÁLISIS Y DISEÑO DEL SISTEMA DE ASIGNACIONES DOCENTES DE LA FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLÁN – MÓDULO DE UAPA	36
4.1.Análisis y requerimientos básicos del usuario.....	36
4.2.Características del sistema.....	39
4.2.1.Hardware y Software.....	39
4.2.2.Struts	41
4.2.2.1.Modelo	42
4.2.2.2.Vista	42
4.2.2.3.Controlador.....	42
4.2.3.Niveles de seguridad	44
4.3.Diseño de la base de datos.....	44
4.3.1.Diagrama de la base de datos	45
4.4.Descripción del módulo de la Unidad de Asuntos del Personal Académico (UAPA)	46
4.5.Pruebas.....	57
4.6.Capacitación.....	58
4.7.Liberación	58
CONCLUSIONES	59
APÉNDICE A	60
BIBLIOGRAFÍA	70

Introducción

La Universidad Nacional Autónoma de México manteniendo su compromiso de mejora continua tanto en el plano académico como en el administrativo, este objetivo se ve reflejado en el plan de desarrollo 2008-2011 y sus líneas rectoras organizadas en seis rubros (docencia, personal académico, investigación, difusión cultural, vinculación y proyección, y gestión y administración universitaria) en las cuales se definen políticas institucionales de mejora continua que permitan el crecimiento en la calidad de los servicios que actualmente brinda la Universidad. Siguiendo la línea de gestión y administración universitaria, la FES Cuautitlán ha puesto gran interés en el desarrollo de sistemas que permitan modernizar y simplificar de manera eficiente los trámites y quehaceres administrativos mediante el apoyo de TI¹ (Tecnologías de la información).

Como parte de este esfuerzo se ha implantado un nuevo sistema informático denominado: **“Sistema de Asignaciones de la Facultad de Estudios Superiores Cuautitlán” (SAFESC)**, que permitirá gestionar de manera segura, eficiente y oportuna las actividades académico-administrativas de la Facultad, el cual se describe en el presente trabajo de tesis conformado por cuatro capítulos, los cuales serán descritos a continuación:

En el capítulo 1 se plantea la situación actual de la FESC, lo que nos permite obtener un panorama general de las actividades que se realizan y los procesos que llevan a cabo, así como los objetivos y metas a desarrollar.

Para cumplir con los objetivos y metas planeados, en el capítulo 2 se explicará brevemente que es un sistema de información y los diferentes modelos de proceso de software que se pueden ocupar para el desarrollo de un sistema. Se describirá que es un sistema distribuido y sus características principales. Asimismo se explica que son las herramientas CASE, su utilidad y las ventajas de utilizarlos en cada una de las etapas de desarrollo de software.

El capítulo 3 menciona las características de los diferentes Sistemas Gestores de Bases de Datos (SGDB), ya que la selección de un SGBD correcto permitirá una mejor comunicación con el lenguaje de programación, así como administración, resguardo y control de la información.

En el capítulo 4 se describe el lenguaje de programación Java, también se explican las actividades que se realizaron para analizar y desarrollar el módulo correspondiente a la Unidad de Asuntos del Personal Académico (UAPA), su implementación y resultados obtenidos. Por último se presentan las conclusiones a las que llego en el desarrollo del sistema.

¹ Tecnologías de la Información (TI). Término utilizado para describir cualquier tecnología que ayuda a producir, manipular, almacenar, comunicar, y/o transmitir información.

Capítulo 1 Antecedentes

1.1.Misión y Visión de la FESC

1.1.1.Misión

La Facultad de Estudios Superiores Cuautitlán, como entidad académica de la UNAM, tiene como función impartir docencia a nivel de licenciatura y posgrado en las áreas de las Ciencias Químicas, Ingenierías, Administración, Agropecuarias y en Artes y Humanidades para instruir, educar y formar profesionales de alto nivel, de fácil inserción laboral, con un claro proyecto de vida y vocación de servicio a su comunidad y al país.

Realiza investigación, que busca contribuir al avance del conocimiento tecnológico y científico, a la solución de retos y problemas de interés regional y nacional. Por sus servicios de extensión y difusión, constituye la mejor propuesta de desarrollo educativo y cultural en su zona de influencia.

Para realizar sus funciones, la Facultad de Estudios Superiores Cuautitlán se ha organizado de forma departamental y matricial con un enfoque multi, inter y transdisciplinario.

Cuenta con profesores e investigadores con formación académica y profesional relevante y pertinente, acorde a las áreas que cultiva y con infraestructura que le permite desarrollar sus actividades sustantivas.

Los principios que guían sus actividades son la libertad de cátedra, la justicia social, la equidad, la creatividad y el liderazgo para el desarrollo educativo de la zona [1].

1.1.2. Visión

La Facultad de Estudios Superiores Cuautitlán es una entidad universitaria reconocida como referente regional, formadora de recursos humanos con actitud emprendedora e innovadora, con compromiso social y alta calidad humana, tanto en sus licenciaturas, como en sus posgrados. Es generadora de conocimientos de vanguardia útiles para la solución de problemas regionales, estatales y nacionales, que a la vez le otorgan recursos. Ha generado políticas de desarrollo interno que la llevan al liderazgo académico dentro de las áreas del conocimiento que en ella se trabajan [2].

1.2.Contexto actual de la FESC

Actualmente en la Facultad de Estudios Superiores Cuautitlán (FESC) son impartidas 15 licenciaturas: Diseño y Comunicación Visual, Químico Farmacéutico Biólogo, Química Industrial, Ingeniería Química, Química, Ingeniería en Alimentos, Contaduría, Administración, Informática, Ingeniería Mecánica Eléctrica, Médico Veterinario Zootecnista, Ingeniería Agrícola, Bioquímica Diagnostica, Farmacia, Tecnología.

Asimismo, se imparten 8 programas de posgrado: Maestría y Doctorado en Ciencias Químicas, Posgrado en Ciencia e Ingeniería de la Computación (Maestría y Doctorado), Maestría y Doctorado en Ingeniería, Maestría y Doctorado en Ciencias de la Producción y de la Salud Animal, que incluye también la Maestría en Medicina Veterinaria y Zootecnia, Posgrado en Ciencias de la Administración (Maestría con orientaciones en Organizaciones y en Finanzas Corporativas), Maestría en Docencia para la Educación Media Superior, Maestría en Artes Visuales y la especialización en Producción de Ovinos y Caprinos.

Durante el semestre 2011-1 la matrícula estudiantil atendida en licenciatura fue de 12,916 alumnos y en posgrado de 205 alumnos.

Actualmente se cuenta con una planta académica de 1,378 profesores los cuales están divididos de la siguiente manera:

Nombramiento	Cantidad
Ayudante de Profesor	163
Profesor de Asignatura	901
Profesor de Carrera	227
Técnico Académico	84
Investigador	2
Emérito	1
TOTAL	1,378

Tabla 1.1. Nombramientos académicos.

Durante el semestre 2010-1 la Facultad tuvo 45 proyectos vigentes del Programa de Apoyo a Proyectos de Institucionales para el Mejoramiento de la Enseñanza (PAPIMES), 23 proyectos del Programa de Apoyo a Proyectos de Investigación e Innovación Tecnológica (PAPIIT), 71 proyectos del Programa de Apoyo a Cátedras de Investigación, Vinculación y Educación (PACIVE) y 16 proyectos con apoyo de CONACYT.

1.3.Organización institucional de la FESC

Para comprender los procesos e interacciones que llevan a cabo las diferentes áreas para el desarrollo de las actividades académico-administrativas es importante conocer como está organizada la FESC, por ello se hace una breve descripción de cómo está organizada.

La FESC está compuesta por la Dirección, Consejo Técnico, 5 Secretarías y 2 Coordinaciones principalmente, de cada Secretaría se desprenden diferentes Coordinaciones o Departamentos que coadyuvan al desarrollo de las actividades del área, el organigrama siguiente muestra de manera general como está estructurada [3]:

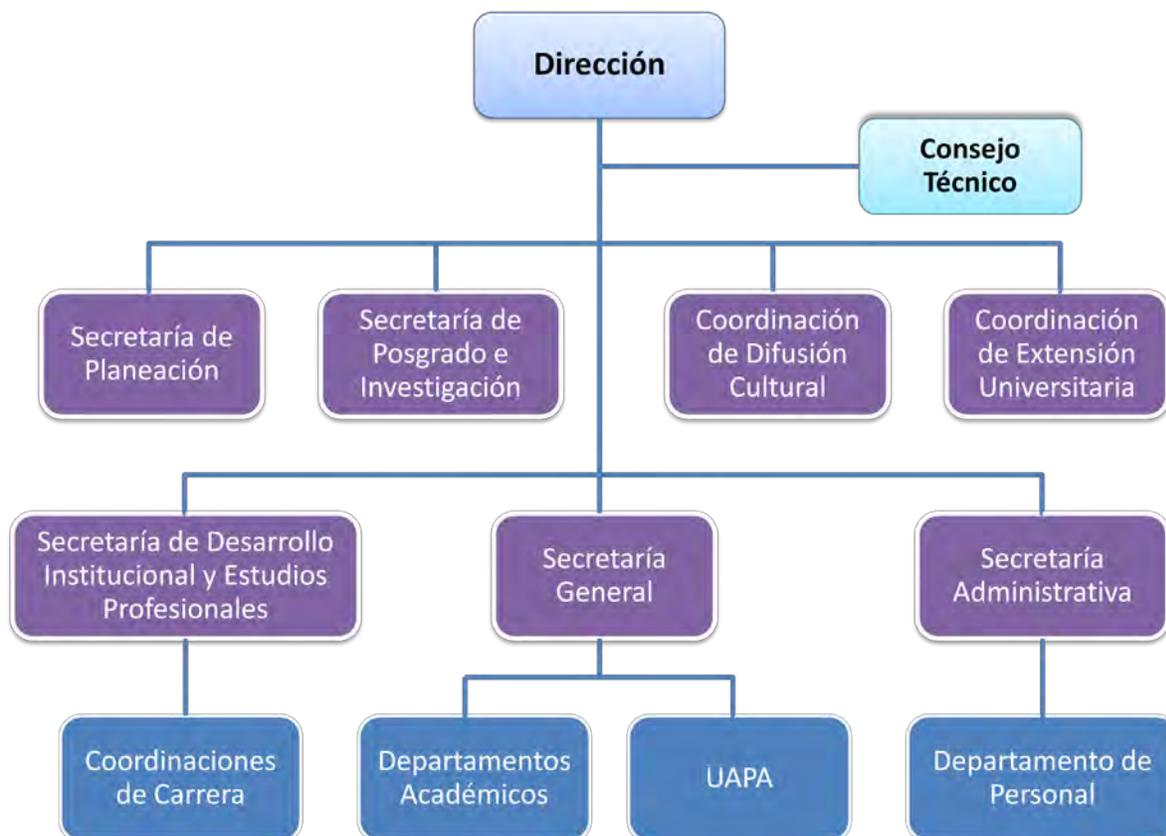


Figura 1.1. Organigrama de la FESC.

Consejo Técnico

El Consejo Técnico es el órgano de autoridad colegiado, de consulta necesaria para los directores. Les corresponde tomar resoluciones en el campo exclusivamente académico para fomentar y reglamentar las actividades de docencia e investigación.

Asuntos que se tramitan, realizan o desahogan:

- Concursos de Oposición abiertos y cerrados
- Ratificación de dictámenes de Concursos
- Recursos de Revisión
- Autorización de Jurados Calificadores

Aprobación de:

- Movimientos y oficios de asignación de los académicos
- Contrataciones y/o movimientos del personal académico de la FESC
- Números de grupos

Secretaría de Planeación:

La Secretaría de Planeación es la encargada de asesorar a la Dirección de la Facultad en materia de planeación, seguimiento y evaluación institucionales, para apoyar y fortalecer el desarrollo de su natural desempeño, asimismo sirve de enlace entre la Facultad y las diferentes instancias de la UNAM, para fortalecer el trabajo en conjunto durante el proceso de planeación institucional.

Las actividades que desarrolla son las siguientes:

- Sistematizar los procesos relativos al acopio, organización y resguardo de la información estadística de la FESC.
- Fortalecer la administración del sistema de plantilla de profesores.
- Elaborar la programación operativa (calendario de actividades académico-administrativas) en la asignación de actividades académicas en coordinación con la Secretaría General, Consejo Técnico, UAPA y La Unidad de Administración Escolar.

Secretaría de Posgrado e Investigación:

La Secretaría de Posgrado e Investigación es la encargada de preparar para la investigación, la docencia y la práctica profesional de alto nivel a hombres y mujeres que desarrollen capacidades de pensamiento complejo y crítico, para la creación y recreación del conocimiento.

Las actividades que desarrolla son:

- Organizar las actividades académicas de los profesores y técnicos académicos adscritos a los diferentes posgrados.
- Realizar las gestiones necesarias para que los exámenes de grado se presenten.

Coordinación de Difusión Cultural:

La Coordinación de Difusión Cultural es el órgano encargado de administrar los recursos materiales y humanos, asignados a realizar labores de extensión y difusión.

Las actividades que desarrolla son:

- Realizar actividades de vinculación con otras instancias generadoras de proyectos culturales, artísticos y científicos.
- Realizar actividades de gestión con instancias que den apoyos o subsidios a programas y proyectos culturales.

Coordinación de Extensión Universitaria:

La extensión universitaria es pieza fundamental en el quehacer de cualquier institución de educación superior, porque contribuye de manera significativa al desarrollo integral de la sociedad haciéndose presente en diversos ámbitos, particularmente en los sectores educativo, cultural, deportivo y recreativo, en ella se deposita la responsabilidad de coadyuvar a la formación integral de los estudiantes, en propiciar el desarrollo armónico y la superación académica de los docentes, en impulsar y proponer alternativas novedosas de acercamiento y extensión del conocimiento y la cultura.

Secretaría de Desarrollo Institucional y Estudios Profesionales:

Su función es la de apoyar a las distintas Coordinaciones de Carrera de la Facultad en su compromiso por mejorar la enseñanza, otorgando condiciones y estableciendo espacios académicos en los que se desarrollen posibilidades reales para una efectiva elevación de la competitividad pedagógica.

Secretaría General:

Su misión es coadyuvar con la Dirección, en la vigilancia, el cumplimiento de la Legislación Universitaria, en los planes y programas de trabajo, y en las disposiciones generales, además de los acuerdos que normen la estructura de la Universidad y de la Facultad.

Coordinar con las jefaturas de departamentos académicos, las propuestas de contratación y remoción del personal académico de la Facultad, conforme a las disposiciones legales establecidas.

Secretaría Administrativa:

La Secretaría Administrativa es un apoyo fundamental para el desarrollo de las actividades de la Facultad, ya que de ella emanan la mayoría de los trámites ante otras instancias internas y externas, los servicios a las instalaciones y a la comunidad que la conforma.

Las actividades que realiza son:

- Acordar con la Dirección de la Facultad las políticas y normas de trabajo que propicien la eficiencia y oportunidad de los distintos servicios y apoyos administrativos confiados a su responsabilidad.
- Gestionar y tramitar los nombramientos, contratos y demás movimientos del personal académico y administrativo adscrito a la Facultad, buscando brindarles la regularización y estabilidad laboral correspondientes.

Las siguientes áreas colaboran en el desarrollo de actividades concernientes al sistema, por ello se hace mención de las mismas y brindar un panorama general de las actividades que se realizan.

Coordinaciones de carrera

Son las áreas encargadas de asesorar a los alumnos en materia de planes de estudio, opciones de titulación, calendario de actividades, asesoría de trámites escolares, servicio social, bolsa de trabajo.

Las actividades que realiza son:

- Planeación de la cantidad de grupos aperturados por semestre.
- Asignación de horarios a los grupos aperturados.
- Enviar a Consejo Técnico la propuesta de grupos para su aprobación.

Departamentos Académicos

Son los responsables de coordinar y promover las actividades académico administrativas que permitan el desarrollo y unificación del personal académico.

Las actividades que realiza son:

- Coordinar con las Jefaturas de Sección y la Unidad de Asuntos del Personal Académico, las propuestas de nuevos ingresos, bajas, nuevos nombramientos, aumentos de horas, etcétera; del personal conforme a los periodos escolares de la UNAM.
- Trabajar con las Coordinaciones de Carrera en cuanto al desarrollo de las actividades cotidianas (planeación de grupos a aperturar en el semestre).

Unidad de Asuntos del Personal Académico (UAPA)

Es la Unidad encargada de brindar un servicio de solución a las solicitudes y dudas que presentan los académicos en cuanto a los trámites que realizan a través del área, cumpliendo con lo establecido en la legislación universitaria. Efectuar los trámites aprobados por las instancias académico-administrativas, cumpliendo con los tiempos marcados en las convocatorias o calendarios establecidos con la finalidad de que el personal académico participe en tiempo y forma.

Las actividades que realiza son:

- Comisiones y Licencias: Realizar los trámites solicitados por el personal académico de la Facultad, para conceder permisos para faltar a sus labores de acuerdo con el Estatuto del Personal Académico (EPA) y al Contrato Colectivo de Trabajo.
- Programa de Apoyo de Superación del Personal Académico (PASPA): Llevar a cabo los trámites correspondientes para que el personal académico goce del beneficio para cumplir con los requisitos que se señalan en el programa.
- Movimientos en la Contratación del Personal Académico: Revisar y analizar los documentos presentados por los Jefes de Departamento para asignar las actividades de docencia directa y apoyo académico al personal de acuerdo a los planes y programas de estudio de la dependencia.
- Estímulos al Personal Académico: Otorgar el reconocimiento al personal académico de asignatura, con base en lo establecido en el Programa de Estímulos a la Productividad y al Rendimiento del Personal Académico de Asignatura (PEPASIG).
- Conceder el reconocimiento al personal académico de carrera, con base en lo establecido en el Programa de Fomento a la Docencia (FOMDOC).

Departamento de Personal

Es el área a la que le corresponde cumplir con los programas, objetivos y metas de la Facultad, mediante el trámite oportuno, registro, control e información de los recursos humanos asignados, además de vigilar su adecuado ejercicio acorde a las políticas institucionales y de la propia dependencia.

Las actividades que realiza son:

- Formular y gestionar ante las dependencias normativas, los movimientos generados por el personal académico y condicionados por el estatuto del personal académico.
- Informar a otras áreas de la dependencia sobre la situación laboral del personal académico.

1.4. Análisis de la problemática

Para el desarrollo de las actividades académico-administrativas, la Facultad cuenta con un calendario anual perfectamente definido, el cual sirve para establecer los tiempos que cada área dispondrá para realizar la parte del trabajo que le corresponde, a continuación se describen paso a paso las tareas que desempeñan cada una de las áreas involucradas en las actividades académico-administrativas y se muestra el calendario:

- Actividad uno:
Las **Coordinaciones de Carrera** realizan una propuesta de grupos con sus respectivos horarios del período anual (ejemplo: 2012-I y 2012-2) y la someten a la aprobación del Consejo Técnico.
- Actividad dos:
En una sesión de **Consejo Técnico** se aprueban la propuesta de grupos o emite las sugerencias necesarias a las Coordinaciones de Carrera para que sus propuestas sean aprobadas. Una vez aprobada la propuesta de grupos, las Coordinaciones de Carrera envían su reporte de grupos a la Unidad de Administración Escolar.
- Actividad tres:
Los **Departamentos y Secciones Académicas** realizan la asignación de Actividades de Docencia y de Apoyo a los nombramientos de los Académicos (a este proceso se le conoce también como asignación de movimientos), además de llevar a cabo lo anterior, recaban toda la información concerniente a la contratación de los académicos. Una vez terminada la asignación de actividades, envía a UAPA la propuesta de movimientos para que a su vez UAPA verifique que la propuesta cumple con normatividad.
- Actividad cuatro:
UAPA recibe un paquete de movimientos por departamento y realiza dos acciones:
 - 1.- Si cumple con la normatividad (de acuerdo al EPA²): envía la propuesta al Consejo Técnico para que sea aprobada.
 - 2.- Si es incorrecta: devuelve la propuesta al Departamento para que realice las modificaciones correspondientes.
- Actividad cinco:

² Estatuto del Personal Académico (EPA). Es el reglamento que regirá las relaciones entre la universidad y su personal académico, en el se establecen los derechos y obligaciones del personal docente.

El **Consejo Técnico** en una de sus sesiones recibe el paquete de movimientos por Departamento Académico y realiza las siguientes acciones:

1.- Aprueba los movimientos y envía la relación de aprobados a UAPA y al Departamento de Personal para que corran los trámites administrativos para su contratación.

2.- Rechaza los movimientos y envía la relación a UAPA para que realice junto con los Departamentos Académicos las modificaciones correspondientes y sea aprobado en una Sesión posterior. En este punto se repiten las actividades del tres en adelante.

Una vez aprobados los movimientos, las Coordinaciones de Carrera vuelven a enviar un reporte de grupos (esta vez incluye a los profesores asignados a sus grupos) a la Unidad de Administración Escolar.

- Actividad seis:

El **Departamento de Personal** recibe por parte de Consejo Técnico el listado de movimientos aprobados y por parte de UAPA y los Departamentos Académicos toda la información necesaria para su contratación del académico (Acta de nacimiento, certificado de estudios, etc.), y empieza a elaborar la forma única³ de contratación del académico e informa a los Departamento Académicos que ya pueden pasar los profesores a firmar su contrato.

³ Forma única: Hoja de contratación que se genera para el personal docente.

PLANEACIÓN DE ACTIVIDADES ACADÉMICO-ADMINISTRATIVAS 2012

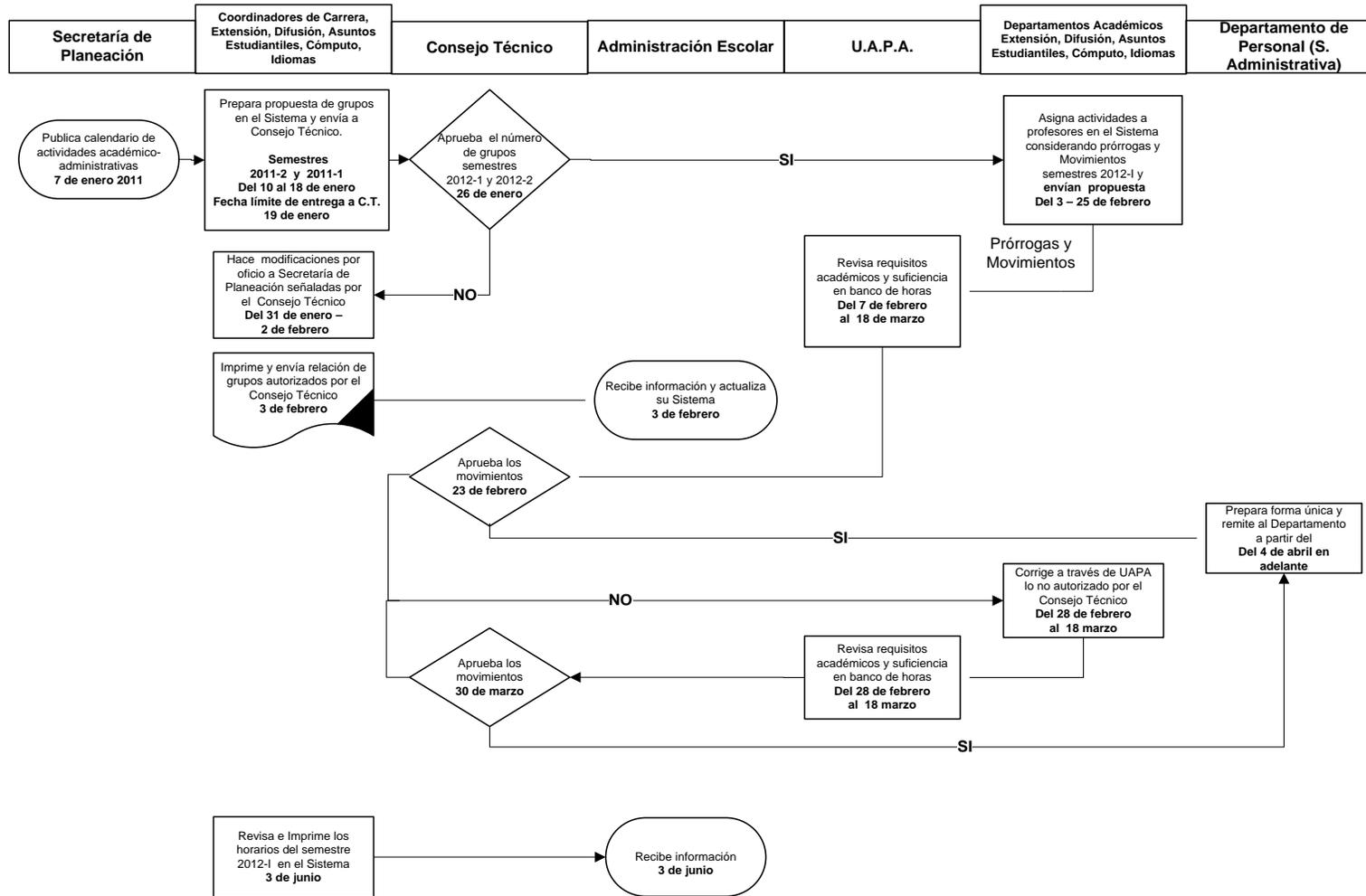


Figura 1.2. Diagrama de flujo de las actividades académico-administrativas.

Para el desarrollo de las actividades anteriores, la Facultad contaba con un sistema que fue implantado en el año 2000 y fue desarrollado mediante Power Builder⁴, este sistema cuenta con las siguientes ventajas y desventajas:

Ventajas:

- Es un sistema ya implantado.
- Control de la aplicación: Al ser una aplicación cliente-servidor solo es accesible por el personal que tenga instalada la aplicación en su equipo.
- Se puede aumentar la cantidad de clientes (aplicaciones instaladas en diferentes equipos).

Desventajas:

- El sistema actual no cuenta con el código fuente para realizar modificaciones o actualizaciones, lo cual significa que este sistema no puede modificarse y en consecuencia no satisface las diferentes necesidades de los usuarios (reportes, nuevas funcionalidades, actualización y corrección de errores).
- Al ser una aplicación cliente-servidor, cada vez que un área sustituya un equipo ó el equipo requiera ser formateado, el programa se tiene que instalar nuevamente, esto genera retrasos en el desarrollo de las actividades de las diferentes áreas.
- EL sistema no se cuenta con documentación técnica ni de usuario: Esto no permite a las áreas documentarse sobre su funcionamiento.

1.5.Propuesta de solución

Una vez realizado el análisis minucioso de los procesos realizados en el desarrollo de las actividades académico-administrativas y el estado del sistema actual se propusieron las siguientes soluciones:

- Hacer una reingeniería del sistema presente para poder recuperar parte del código fuente y tratar de actualizarlo para responder a las necesidades actuales de la FESC, este proceso involucra personal capacitado e invertir tiempo para analizar la forma de trabajar del sistema actual antes de poder hacer alguna actualización, así mismo requiere una evaluación para ver si es viable su escalabilidad.
- Desarrollar un nuevo sistema en línea que permita el trabajo desde cualquier equipo que se encuentre dentro de la FESC, utilizando las tecnologías actuales, permitiendo ajustarnos a las necesidades presentes y una mayor escalabilidad.
- Mantener el sistema actual.

⁴ PowerBuilder. Es una herramienta de desarrollo de aplicaciones informáticas desarrollada por la empresa Sybase.

Una vez analizadas las propuestas anteriores se optó por el desarrollo de un nuevo sistema en línea, ya que nos permitirá tener más ventajas a corto y largo plazo como se describe a continuación:

- Se realizará un análisis a los procesos para mejorarlos y hacerlos más eficientes.
- Se contará con el código fuente, esto permitirá realizar, actualizaciones y agregar nuevas funcionalidades que se adapten a las necesidades que surjan con el paso del tiempo.
- Se contará con manuales técnicos y de usuarios finales.
- Se ahorrará tiempo en el desarrollo de las actividades de cada área que utiliza el sistema ya que será más eficiente.
- Alta disponibilidad de la información ya que no se tendrá que instalar la aplicación en cada equipo utilizado por los usuarios, evitando contratiempos al adquirir un nuevo equipo ó cuando es formateado.

1.6.Objetivos

1.6.1. Objetivos generales

- Proporcionar a la FESC un sistema en línea para la gestión de las actividades académico-administrativas, permitiendo elevar la eficiencia en el desempeño de sus tareas cotidianas y reducir los tiempos de ejecución. Así como el apropiado uso de la tecnología en el desarrollo del sistema, para permitir su posterior mantenimiento, actualización y escalabilidad.

1.6.2. Objetivos específicos

- Proporcionar información oportuna a todas las áreas involucradas con el sistema que la requieran y que tengan los privilegios.
- Facilitar a las Coordinaciones de Carrera crear su propuesta de grupos y sus horarios para que sean propuestos al Consejo Técnico.
- Facilitar a los Departamentos y Secciones Académicas la asignación de actividades de docencia y de apoyo a los nombramientos de los académicos.
- Proporcionar a UAPA un sistema que permita evaluar de manera eficiente las propuestas de los Departamentos, así como el envío de las mismas a Consejo Técnico para su posterior aprobación ó seguimiento.
- Proporcionar al Consejo Técnico un sistema que permita aprobar la asignación de movimientos a los nombramientos de los académicos.
- Proporcionar al Departamento de Personal un sistema que le permita dar seguimiento a la contratación del personal académico.
- Facilitar a la administración de la Facultad la gestión de los servicios que proporciona el sistema mediante el control de privilegios de los usuarios.

1.7.Metas

- Reducir en un 50% el tiempo que tardan las Coordinaciones en crear la propuesta de grupos con sus horarios.
- Reducir en un 30% el tiempo que tardan los Departamentos Académicos en asignar las actividades de docencia y apoyo a los nombramientos de los académicos.
- Reducir en un 50% el tiempo que UAPA dedica a revisar los movimientos propuestos por los Departamentos Académicos.
- Hacer que el tiempo de aprobación de movimientos por parte del Consejo Técnico se disminuya en un 50%.

Una vez conocido el contexto de la FESC y establecida la problemática actual en el desarrollo de las actividades académico-administrativas, se planteó como solución el desarrollo de nuevo sistema en línea.

Existen diferentes tipos de software y también diferentes modelos que sirven de apoyo para el desarrollo de los mismos, por ello es necesario conocer los diferentes modelos existentes para utilizarlos de manera individual ó combinados para la desarrollo del software que se requiere.

El siguiente capítulo muestra algunos modelos de desarrollo de software que se tomaron en cuenta en el análisis y desarrollo de este caso práctico.

Capítulo 2 Sistemas de información y sistemas distribuidos

El constante desarrollo de las tecnologías de la información (TI) ha cambiado radicalmente la forma en cómo operan las organizaciones, ya que el uso de las mismas permite automatizar los procesos operativos, brinda un soporte para el análisis, resguardo y tratamiento de la información, así como permitir su posterior uso en la toma de decisiones, dando como resultado disminución de tiempos y costos; por lo tanto las TI brindan un valor agregado a las organizaciones que establecen ventajas competitivas.

2.1.¿Qué es un sistema de información?

Existen tantas definiciones de sistemas de información como autores que hablan sobre el tema, es importante mencionar que aunque cada autor maneja su propia definición todas ellas tienen puntos en común, a continuación menciono algunas de ellas:

“Conjunto de personas, datos, procesos y tecnología de la información que interactúan para recopilar, procesar, guardar y proporcionar como salida la información necesaria para brindar soporte a una organización” (Whitten Bentley, 2011)[4]

“Un sistema de información es un sistema cuyo propósito es gestionar y dar acceso a una base de datos de información. Los conflictos en los sistemas de información incluyen seguridad, usabilidad, privacidad y mantenimiento de la integridad de los datos” (Sommerville, 2011)[5]

Como se puede observar existen varios factores en común en las definiciones antes mencionadas, de lo cual podemos deducir que un sistema de información es un conjunto de elementos (tanto humanos como tecnológicos) interrelacionados con el fin de analizar, procesar y gestionar los datos de una organización para entregar como resultado información que sea útil para la toma de decisiones.

2.2.Ciclo de vida de software

En un principio al desarrollar e incorporar un sistema de información en una organización, la persona encargada de diseñar/planear este sistema realizaba un levantamiento de las necesidades antes de comenzar a construirlo, seguidamente comenzaba la programación del mismo, pero esta tarea no estaba administrada, supervisada o gestionada de ningún modo, por lo tanto se iba corrigiendo a medida que surgían los errores tanto de su programación o por el cambio de requerimientos solicitados por el cliente o usuario final; con el paso del tiempo los sistemas comenzaron a ser más robustos, aumentado así su complejidad y por lo tanto, no se tenía control sobre los cambios realizados al programa, surgió entonces la necesidad de tener un ciclo de vida de este tipo de proyectos que permitiera cumplir las necesidades o especificaciones, no sólo las que dieron la necesidad del programa, sino

también aquellas que fueron surgiendo sobre la marcha, así es como nace el concepto de ciclo de vida de software.

La ISO (**International Organization for Standardization**) en su norma ISO/IEC 12207 define al ciclo de vida de un software como un marco de referencia que contiene las actividades y las tareas involucradas en el desarrollo, explotación y el mantenimiento de un producto de software, abarcando desde la definición hasta la finalización de su uso[6].

Por lo tanto podemos decir que el ciclo de vida de software es una metodología⁵ que busca guiarnos de manera sistemática en la gestión y administración de un proyecto con el fin de llevarlo a cabo con un alto grado de calidad y éxito. Dentro del ciclo de vida de software se tienen seis actividades básicas [5], las cuales se mencionan a continuación:

- Investigación preliminar.
- Determinación de los requerimientos del sistema.
- Diseño del sistema.
- Desarrollo de software.
- Prueba de los sistemas.
- Implantación y evaluación.

Investigación preliminar

Cuando se realiza la petición de un sistema de información, se da comienzo a la primera actividad que se divide en tres partes:

- Aclaración de la solicitud: Muchas de las solicitudes que provienen de los empleados y usuarios no están formuladas de manera clara. Por consiguiente, antes de considerar cualquier investigación de sistemas, la solicitud debe examinarse para determinar con exactitud lo que desea el solicitante.
- Estudio de factibilidad: Un resultado importante de la investigación preliminar es llegar a la determinación de que el sistema solicitado sea factible. Existen tres aspectos relacionados con el estudio de factibilidad:
 - Factibilidad técnica: Es la evaluación técnica donde se plantea si se puede desarrollar el software con el equipo actual, la tecnología de software y el personal disponible o si se requieren más recursos.
 - Factibilidad económica: Es la evaluación de los beneficios que se obtienen con el nuevo software, para aceptar los costos de desarrollo o aceptar los costos asociados de no crear el sistema.

⁵ Metodología: Conjunto de pasos o procedimientos que se emplean para alcanzar un objetivo.

- Factibilidad operacional: Es donde se analiza si se desarrolla e implanta el sistema, ¿qué resistencia al cambio se tendrá por parte de los usuarios ó si será usado el sistema?
- Aprobación de la solicitud: Dependiendo de los resultados del estudio de factibilidad, de si es deseable y factible el desarrollo del software, se aprobará la solicitud.

Determinación de los requerimientos del sistema

En esta actividad se determinan los procesos que lleva a cabo una empresa, y se realiza mediante entrevistas a los empleados y administradores de la organización, entre las preguntas clave que se deben desarrollar en la entrevista se encuentran:

- ¿Qué es lo que se hace?
- ¿Cómo se hace?
- ¿Con qué frecuencia se presenta?
- ¿Qué tan grande es el volumen de transacciones o de decisiones?
- ¿Cuál es el grado de eficiencia con el que se efectúan las tareas?
- ¿Existe algún problema?
- Si existe algún problema, ¿qué tan serio es?
- Si existe algún problema ¿cuál es la causa que lo origina?

Además de los cuestionarios, también se requiere el estudio de los manuales, reportes y documentos existentes para comprender los procesos en su totalidad. Conforme se reúnen los detalles, se van identificando las características que debe tener el nuevo sistema, incluyendo la información que debe producir como las características operacionales, controles de procesamiento, tiempos de respuesta, métodos de entrada y salida.

Diseño del sistema

Una vez establecidas las especificaciones del software a desarrollar se procede a crear el diseño del sistema, que puede ser representado de muchas maneras (diagramas, tablas, símbolos, etc.), esto proporcionará al equipo de programadores la información necesaria (especificaciones, datos de entrada o salida, procedimientos, etc.).

Desarrollo de software

En esta actividad se procede al desarrollo del sistema como tal y los encargados de realizarla tienen la obligación de documentar y proporcionar la información necesaria de cómo y por qué ciertos procedimientos se realizaron en determinada forma, ya que esta documentación es esencial para llevar a cabo el mantenimiento del sistema una vez que sea instalado.

Pruebas del sistema

Durante esta actividad se realizan una serie de pruebas al sistema (pruebas de instalación, rendimiento, estrés, etc.) para ver que realmente cumple las necesidades del cliente, algunas de las pruebas son realizadas por el equipo de desarrollo del software y otras deben ser

realizadas con el usuario final, ya que esto puede ayudar a detectar operaciones no previstas y darles solución lo antes posible. Las pruebas también pueden ser realizadas por personas ajenas al grupo que programó el sistema, pero con un alto conocimiento en el desarrollo de sistemas, garantizando que las pruebas sean completas e imparciales.

Implantación y evaluación

En esta actividad se instala el sistema, mediante cierta estrategia que permita la utilización del mismo con el menor riesgo posible, esto estará a consideración de la organización y su situación, también se debe tener un plan de capacitación para los usuarios que harán uso del mismo.

Una vez instalado el sistema se establecerá un plan de mantenimiento que permita realizar los cambios o modificaciones al software para satisfacer las nuevas necesidades de los usuarios, dado que las organizaciones experimentan cambios, los sistemas de información deben de mantenerse al día.

Dentro del ciclo de vida de software se encuentran distintos modelos, los cuales no son descripciones definitivas de los procesos de desarrollo de software, son abstracciones de los procesos que se pueden utilizar como guía al momento de realizar un proyecto, la elección de uno u otro dependerá de las características del proyecto y los acuerdos que se tengan entre los desarrolladores y el cliente, a continuación se mencionan algunos modelos.

2.3. Modelos de ciclo de vida

2.3.1. Modelo en cascada

Es el primer modelo que se publicó por Royce en 1970 [7], las principales etapas de este modelo se transforman en actividades fundamentales de desarrollo:

1.- Análisis y definición de requerimientos:

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

2.- Diseño del sistema y del software:

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

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 con su especificación.

4.- Intergración y prueba del sistema:

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

5.- Funcionamiento y mantenimiento:

Por lo general, esta 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 los nuevos requerimientos.

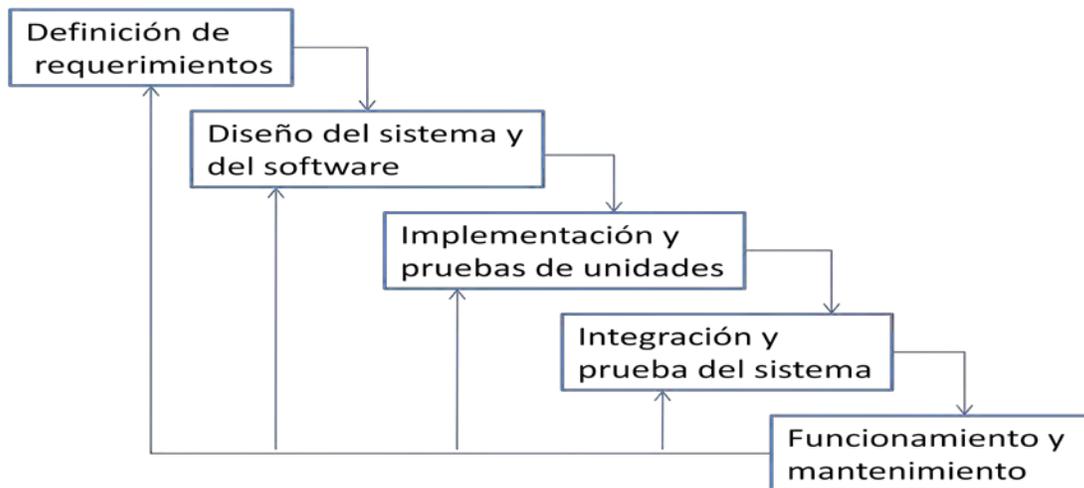


Figura 2.1. Modelo de ciclo de vida en cascada.

2.3.2. Modelo de desarrollo evolutivo

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.



Figura 2.2. Modelo de ciclo de vida de desarrollo evolutivo.

Este modelo es utilizado en desarrollos donde los requisitos de los usuarios suelen cambiar conforme se desarrolla el software, por lo que se entrega una primera versión funcional limitada y posteriormente se seguirán entregando nuevas versiones hasta llegar al objetivo deseado.

2.3.3. Modelo de desarrollo basado en componentes

En la mayoría de los proyectos de software existe algo de reutilización de software. Por lo general esto sucede con las personas que trabajan en el proyecto y conocen diseños de código similares al requerido, modifican el código existente y lo incorporan al nuevo sistema.



Figura 2.3. Modelo de ciclo de vida de desarrollo basado en componentes.

2.3.4. Modelo de desarrollo en espiral

Este modelo se divide en un número de actividades llamadas regiones de tareas, en general existen 6 regiones:

- Región 1: Tareas requeridas para establecer la comunicación entre el cliente y el desarrollador.
- Región 2: Tareas relacionadas a la definición de recursos, tiempo e información relacionada con el proyecto.
- Región 3: Tareas necesarias para evaluar los riesgos técnicos y de gestión del proyecto.
- Región 4: Tareas para construir la o las representaciones del sistema.
- Región 5: Tareas para construir la aplicación, instalarla, probarla y proporcionar soporte al cliente.
- Región 6: Tareas para obtener la reacción del cliente, según la evaluación de lo creado e instalado en ciclos anteriores.

Al inicio del ciclo de este modelo el equipo de desarrollo de software comenzará por el centro y en el sentido que se indica en la figura siguiente:

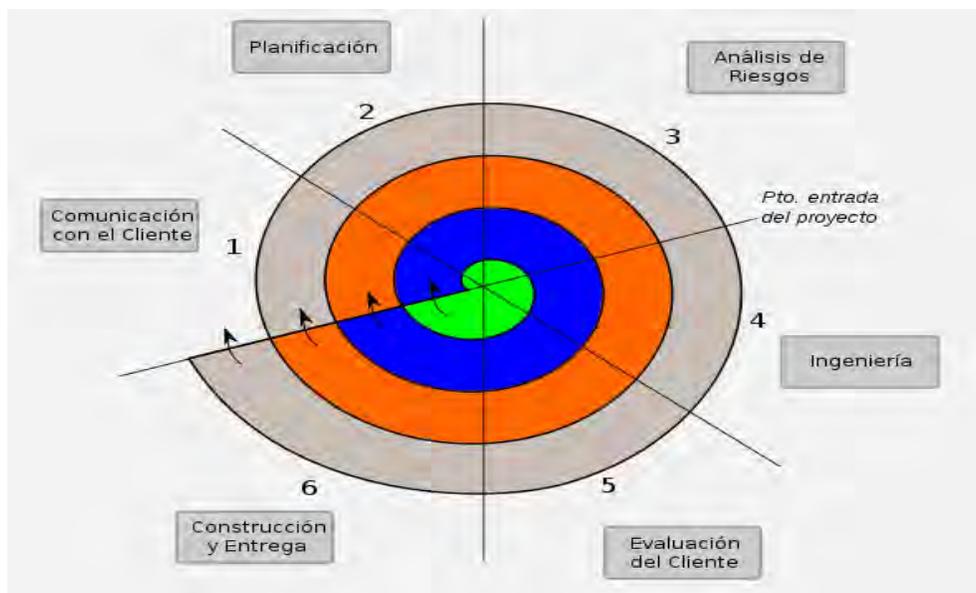


Figura 2.4. Modelo de ciclo de vida de desarrollo en espiral.

En el primer circuito de la espiral se puede producir el desarrollo de una especificación del producto, los pasos siguientes podrían generar un prototipo y progresivamente versiones más sofisticadas del software. Cada paso por la región de planificación provoca ajustes en el plan del proyecto, los costes y planificación se realimentan en función de la evaluación del cliente. Las actividades mencionadas se manejan para cualquier tipo de desarrollo de software (desarrollos pequeños o grandes), la única diferencia que tendrán son la cantidad de tareas que se realizaran en cada etapa, ya que un desarrollo robusto tendrá más actividades que realizar.

2.4. Sistemas distribuidos

2.4.1. Definición de sistemas distribuidos

Un sistema distribuido se define como una colección de computadoras separadas físicamente y conectadas entre sí por una red de comunicaciones, se comunican y coordinan sus acciones mediante el paso de mensajes, para el logro de un objetivo. Cada equipo posee sus componentes de hardware y software que el usuario percibe como un solo sistema (no necesita saber qué cosas están en qué máquina), el usuario accede a los recursos remotos de la misma manera en como accede a los recursos locales [9][10].

Un sistema distribuido es *“una colección de computadoras independientes que aparecen al usuario como un solo sistema coherente”* Tanenbaum y Van Steen (2007) [7].

Existen gran cantidad de aplicaciones para los sistemas distribuidos que van desde la provisión de capacidad de cómputo a grupos de usuarios, sistemas bancarios, comunicaciones multimedia, así como aplicaciones comerciales y técnicas. Las características que debe cumplir una aplicación que sea distribuida son:

- **Recursos compartidos:** Un sistema distribuido permite compartir los recursos de hardware y software, tales como discos, impresoras, archivos y compiladores que se asocian con computadoras en una red.
- **Apertura:** Los sistemas distribuidos, por lo general, son sistemas abiertos, lo cual significa que están diseñados entorno a protocolos estándar que permiten la combinación de equipo y software de diferentes proveedores.
- **Concurrencia:** Esta característica de los sistemas distribuidos permite que los recursos disponibles en una red puedan ser utilizados simultáneamente por los usuarios y/o agentes que interactúan en la red.
- **Escalabilidad:** Al menos en principio, los sistemas distribuidos son escalables cuando las capacidades del sistema pueden aumentarse al agregar nuevos recursos para enfrentar las nuevas demandas del sistema.
- **Tolerancia a fallas:** En el caso de los sistemas distribuidos, puede ser que falle un componente o parte de un componente, pero es posible que el resto del sistema continúe sus funciones, dependiendo de los planes de contingencia que se establezcan los sistemas serán más tolerantes a las fallas tanto de hardware (con el uso de componentes redundantes) como de software (diseño de programas que sean capaces de recuperarse a los fallos) [11][12].

2.4.2. Modelo cliente-servidor

En un sistema cliente-servidor, el usuario interactúa con un programa que se ejecuta en su computadora (por ejemplo, un navegador Web o una aplicación instalada localmente). Éste cliente interactúa con otro programa que se ejecuta en una computadora remota (por ejemplo, un servidor Web o un gestor de base de datos). La computadora remota proporciona servicios, como acceso a páginas Web, que están disponibles a clientes externos [7].

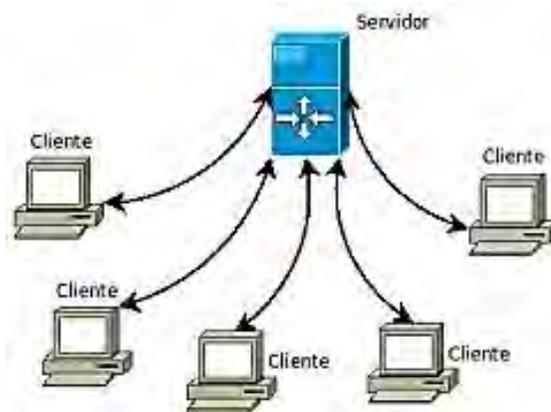


Figura 2.5. Modelo cliente-servidor.

La separación entre cliente y servidor es una separación lógica, donde el servidor no se ejecuta necesariamente sobre una sola máquina ni es necesariamente un sólo programa. Los tipos específicos de servidores incluyen los servidores web, servidores de archivo, servidores de correo; Mientras que sus propósitos varían de unos servicios a otros, la arquitectura básica seguirá siendo la misma [12].

Cliente

El cliente es el proceso que permite al usuario formular los requerimientos y pasarlos al servidor, se le conoce con el término *front-end*.

El Cliente normalmente maneja todas las funciones relacionadas con la manipulación y despliegue de datos, por lo que están desarrollados sobre plataformas que permiten acceder a los servicios distribuidos en cualquier parte de una red.

Las funciones que lleva a cabo el proceso cliente se resumen en los siguientes puntos:

- Administrar la interfaz de usuario.
- Interactuar con el usuario.
- Procesar la lógica de la aplicación y hacer validaciones locales.
- Generar requerimientos de bases de datos.
- Recibir resultados del servidor.
- Formatear resultados.

Servidor

Es el proceso encargado de atender a múltiples clientes que hacen peticiones de algún recurso administrado por él. Al proceso servidor se le conoce con el término back-end. El servidor normalmente maneja todas las funciones relacionadas con la mayoría de las reglas del negocio y los recursos de datos.

Las funciones que lleva a cabo el proceso servidor se resumen en los siguientes puntos:

- Aceptar los requerimientos de bases de datos que hacen los clientes.
- Procesar requerimientos de bases de datos.
- Formatear datos para transmitirlos a los clientes.
- Procesar la lógica de la aplicación y realizar validaciones a nivel de bases de datos.

2.4.3. Arquitectura de dos capas

La arquitectura básica del modelo cliente servidor es la arquitectura de dos capas y consta de 3 componentes distribuidos en dos capas: el cliente y el servidor.

Los tres componentes son:

- Interfaz de usuario.
- Gestión del procesamiento.
- Gestión de la base de datos.

Existen 3 tipos de arquitecturas cliente servidor de dos capas:

- Clientes obesos.
- Clientes delgados.
- Clientes híbridos.

Cliente pesado (Servidor delgado)

Un cliente pesado tiene capacidad de almacenar los datos y procesarlos, pero sigue necesitando las capacidades del servidor. Un cliente de correo electrónico suele ser un cliente pesado. Puede almacenar los mensajes de correo electrónico del usuario, trabajar con ellos y redactar nuevos mensajes, pero sigue necesitando una conexión al servidor para enviar y recibir los mensajes.

Cliente delgado (Servidor pesado)

Un cliente liviano no tiene capacidad de procesamiento y su única función es recoger los datos del usuario, dárselos al servidor, y mostrar su respuesta. Los primeros navegadores web eran clientes livianos, simplemente mostraban las páginas web que solicitaba el usuario. Actualmente, el uso de lenguajes de script, programas Java y otras funciones de DHTML dan una capacidad de procesamiento a los navegadores, por lo que se consideran clientes Híbridos.

Cliente Híbrido

Un cliente híbrido no tiene almacenados los datos con los que trabaja, pero sí es capaz de procesar datos que le envía el servidor. Muchos programas de colaboración almacenan remotamente los datos para que todos los usuarios trabajen con la misma información, y utilizan clientes híbridos para acceder a esa información [13].

2.4.4. Arquitectura de tres capas

La arquitectura cliente-servidor de tres capas es una extensión de la arquitectura cliente-servidor de dos capas que pretende el desacoplamiento entre el cliente y el servidor mediante una capa intermedia (middle tier), el reparto de funciones se lleva de la siguiente manera: una capa para la presentación (interfaz del usuario), otra para cálculos y procesamiento (donde se encuentra el modelo de negocio) y otra para el almacenamiento (acceso a los datos), en donde una capa solo tiene relación con la siguiente, comúnmente cada capa se encuentra en una plataforma diferente [14][15][16][17].

Características:

- Centraliza la gestión de la reglas del negocio en un único lugar.
- Los clientes piden o envían información a esta aplicación centralizada, no al gestor de base de datos en el servidor.

Ventajas

- Los componentes de la aplicación pueden ser desarrollados en cualquier lenguaje general lo que posibilita que el grupo de desarrolladores no se centre en el uso de un solo lenguaje.
- Los componentes están centralizados lo que posibilita su fácil desarrollo, mantenimiento y uso.
- Los componentes de la aplicación pueden estar esparcidos en múltiples servidores permitiendo una mayor escalabilidad.

Los problemas de limitación para las conexiones a las bases de datos se minimizan ya que la base de datos sólo es vista desde la capa intermedia y no desde todos los clientes. Además de que las conexiones y los drivers de las bases de datos no tienen que estar en los clientes.



Figura 2.6. Modelo cliente-servidor tres capas.

2.4.5. Arquitectura multi-capas

En la arquitectura cliente-servidor multi-capa la presentación, la lógica de negocio y la gestión de los datos son procesos lógicamente separados. Una regla fundamental en una arquitectura multi-capa es que el nivel de cliente no se comunica directamente con el nivel de datos, todas las comunicaciones deben pasar por el nivel intermedio, eso ocurre para mejorar el balance la carga en los diversos servidores permitiendo una mayor escalabilidad.

El uso más común de la arquitectura multi-capa es la arquitectura de tres capas.

Ventajas

- **Centralización del control:** Los accesos, recursos y la integridad de los datos son controlados por el servidor de forma que un programa cliente defectuoso o no autorizado no pueda dañar el sistema.
- **Escalabilidad:** Se puede aumentar la capacidad de clientes y servidores por separado. Cualquier elemento puede ser aumentado (o mejorado) en cualquier momento.
- **Fácil mantenimiento:** Al estar distribuidas las funciones y responsabilidades entre varios equipos independientes, es posible reemplazar, reparar, actualizar, o incluso trasladar un servidor, mientras que sus clientes no se verán afectados por ese cambio (o se afectarán mínimamente). [13][18].

2.5. Herramientas CASE⁶

El crecimiento de las tecnologías de información permitió crear sistemas más robustos, pero dada la complejidad para hacer estos sistemas se tuvo la necesidad de crear herramientas que permitieran asistir en el desarrollo de las diferentes etapas que tiene el ciclo de desarrollo del software (investigación preliminar, análisis, diseño, desarrollo, pruebas e implementación), a este tipo de herramientas se les conoce como **”Herramientas CASE”**, algunas de las ventajas que ofrecen son:

1. Mejorar la productividad en el desarrollo y mantenimiento del software.
2. Aumentar la calidad del software.
3. Mejorar el tiempo y coste de desarrollo y mantenimiento de los sistemas informáticos.
4. Mejorar la planificación de un proyecto.
5. Aumentar la biblioteca de conocimiento informático de una empresa ayudando a la búsqueda de soluciones para los requisitos.
6. Automatizar, desarrollo del software, documentación, generación de código, pruebas de errores y gestión del proyecto.
7. Ayuda a la reutilización del software, portabilidad y estandarización de la documentación
8. Gestión global en todas las fases de desarrollo de software con una misma herramienta.
9. Facilitar el uso de las distintas metodologías propias de la ingeniería del software.

Las herramientas case más ocupadas en bases de datos son:

- Erwin.
- EasyCASE.
- Oracle Designer.
- DBDesigner.

⁶ *Computer Aided Software Engineering*, Ingeniería de Software Asistida por Computadora

Capítulo 3 Sistemas Gestores de Bases de Datos (SGBD)

3.1. Bases de datos

El término de bases de datos fue escuchado por primera vez en 1963, en un simposio celebrado en California–Estados Unidos[19]. Una base de datos se puede definir como un conjunto de información relacionada que se encuentra agrupada ó estructurada. Actualmente el uso de las bases de datos en cualquier empresa es imprescindible, ya que el uso de las mismas aporta muchas ventajas en el manejo de la información [20].

Conceptos básicos de las bases de datos:

Dato:

Se puede decir que un dato es “Todo hecho aislado y en bruto que por sí solo no representa nada, el cual debe ser procesado dentro de algún contexto para obtener de él resultados relacionados con la evaluación e identificación de personas, eventos y objetos”[20].

En computación también es la unidad mínima de información.

Información:

La información es un conjunto de datos ya procesados e interrelacionados que tienen un significado, la cual nos sirve para tomar decisiones en un determinado contexto.

Análisis de la información:

En bases de datos se puede definir como un modelo de datos que consiste en la representación conceptual un problema que se desea resolver, y cuya característica principal es la claridad de su contenido.

Definición de una base de Datos:

Son un conjunto de datos almacenados y relacionados entre sí con un objetivo en común, los cuales se encuentran ordenados y clasificados. Se puede entender también como base de datos al lugar físico donde se encuentran almacenados los datos.

3.1.1. Arquitectura de las Bases de Datos

Las bases de datos manejan la arquitectura de tres niveles definida por el grupo ANSI/SPARC [21] [22].

La definición de cada uno de esos niveles es la siguiente:

Nivel interno o físico:

Es el nivel más bajo de abstracción y es donde se define como se almacenan los datos físicamente, así como los métodos de acceso.

Nivel conceptual:

Es el nivel medio de abstracción en el cual se pretende representar de forma total los datos que componen las bases de datos, es decir es la visión organizativa total de los datos contenidos dentro de ella.

Nivel externo:

Este es el nivel más alto de abstracción en el cual se muestra una visión parcial de los datos para un usuario o grupo de usuarios, es decir solo muestra ciertos datos en particular para usuarios en específico.

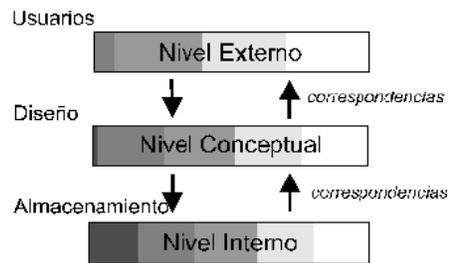


Figura 3.1 Arquitectura de tres niveles de las bases de datos.

3.1.2. Características de las bases de datos

Entre las principales características [23][24] que debe cumplir una base de datos están:

Redundancia:

La redundancia se refiere a la existencia de información repetida o duplicada en la base de datos, la redundancia acarrea muchos problemas con la integridad y consistencia de los datos almacenados, ya que afecta en los procesos de entrada, actualización, y eliminación de los mismos. Dentro de las bases de datos relacionales puede existir redundancia pero debe ser mínima y controlada.

Consistencia:

Como su nombre lo describe la consistencia se refiere a que algo sea constante. En las bases de datos quiere decir que si existe redundancia y no es controlada, al momento de acceder a los datos, actualizarlos, modificarlos o eliminarlos, se corre el riesgo de no realizarlo totalmente, produciendo errores y generando datos inconsistentes; por ello para que las bases de datos conserven su consistencia deben garantizar que no exista redundancia o que sea mínima y controlada.

Integridad:

La integridad en las bases de datos se refiere a que debe existir consistencia en los datos y además que el valor contenido en los datos sean válidos de acuerdo a las políticas que se tengan.

Seguridad:

La seguridad en las base de datos se refiere a garantizar un control de acceso, modificación y definición, tanto de los datos como de la estructura de la propia base de datos, por parte de los diferentes usuarios.

Estos conceptos se encuentran interrelacionados, ya que al no cumplirse uno de ellos los demás no pueden cumplirse.

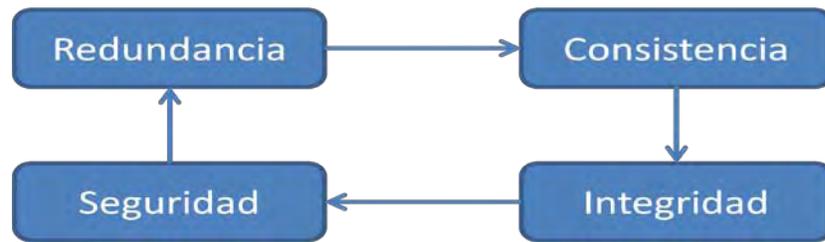


Figura 3.2 Características de las bases de datos.

3.1.3. Modelo de datos

Para definir que es un modelo de datos empezaremos primero definiendo que es un modelo:

Modelo:

Es una representación conceptual de la realidad la cual contiene las características generales del objeto, proceso o acciones que se va a realizar. En bases de datos esto se logra de forma gráfica.

Modelo de datos:

Un conjunto de herramientas conceptuales que nos van a servir para describir los datos, así como las relaciones existentes entre ellos y la semántica asociada a ellos.

Los modelos de datos se dividen en tres principales grupos:

Modelos lógicos basados en objetos.

En este tipo de modelos se encuentran en los niveles conceptual y externo, es decir representan los datos de forma en cómo los captamos del mundo real, en este grupo se encuentra el modelo entidad-relación.

Modelos lógicos basados en registros.

Este grupo de modelos se encuentran en el nivel conceptual y físico, en este tipo de modelos ocupan registros y estados, para poder representar la información, y sus relaciones se manejan por medio de ligas o apuntadores, en este grupo se encuentran:

- Modelo jerárquico.
- Modelo de red.
- Modelo relacional.
- Modelo orientado a objetos.

Modelos físicos de datos.

En este tipo de modelos se representan aspectos de implementación de la base de datos, en este grupo se encuentran:

- Modelo unificador.
- Memoria de sistemas.

3.1.4. Modelo relacional

Los datos se representan en el modelo relacional por medio de tablas también llamadas relación, las cuales contiene filas (tuplas) y columnas (atributos), de esta forma una tupla es un registro y las columnas son los atributos que contiene una tupla [23][24].

La siguiente tabla muestra las relaciones entre los elementos de la tabla y sus nombres dentro del modelo relacional:

RELACIÓN	TABLA
Tupla	Fila
Atributo	Columna

Figura 3.3 Relación entre elementos del modelo relacional.

Claves

Las asociaciones entre las diferentes tablas o relaciones se llevan a cabo mediante una clave o llave, la cual es un conjunto de atributos no vacíos que identifican de manera única a una tupla.

Existen básicamente tres tipos de claves o llaves y son los siguientes:

Llave prima: Es aquella clave que permite identificar a una tupla de manera única en una relación.

Llaves foráneas: Es la clave conformada por un conjunto de atributos no vacíos (puede ser solo uno) dentro de una tabla, los cuales deben corresponder con el valor de la clave primaria de otra tabla [25].

3.1.5. Bases de datos relacionales

Una base de datos es un conjunto de datos relacionados dentro de un contexto, los cuales se encuentran agrupados y estructurados, las bases de datos relacionales tienen el mismo fin, pero con características especiales como:

- El que representan a la información mediante tablas, las cuales ofrecen una manera más fácil de acceder a los datos que se encuentran dentro de ellas.
- Una tabla solo contiene un número limitado de campos.
- Cada registro dentro de la tabla es distinto.

3.1.6. Reglas de Codd⁷

En la década de los 80 comenzaron a aparecer numerosos Sistemas Gestores de Bases de Datos (SGBD) los cuales se hacían pasar por relacionales simplemente por almacenar su información en tablas, perdiendo así las características del modelo relacional como el manejo de claves primarias, etc.

Las 12 reglas de Codd son un conjunto de reglas que determinan la fidelidad de un SGBD relacional al modelo relacional, esto quiere decir que entre más reglas cumpla será más apegado a un modelo relacional.

⁷ En 1984 Codd publicó 12 reglas que un verdadero sistema relacional debería de cumplir. En la práctica algunas de ellas son difíciles de realizar. Un sistema podrá considerarse "más relacional" cuanto más siga estas reglas.

0. Cualquier SGBD que proclame ser relacional, deberá manejar, completamente, las bases de datos por medio de sus capacidades relacionales.

1. Regla de información:

Toda la información dentro de una base de datos relacional se representa de manera explícita a nivel lógico y exactamente de una sola manera, como valores en una tabla.

2. Regla del acceso garantizado:

Se garantiza que todos y cada uno de los datos (valor atómico) en una base de datos relacional pueden ser leídos recurriendo a una combinación del nombre de la tabla, valor de la llave primaria y nombre de la columna.

3. El manejo sistemático de los valores nulos:

En un DBMS totalmente relacional se soportan los valores nulos (que son distintos de una cadena de caracteres vacía o de una cadena con caracteres en blanco o de cero o cualquier otro número), para representar información faltante o no aplicable de una forma consistente, independientemente del tipo de dato.

4. Catálogo dinámico en línea basado en un modelo relacional:

La descripción de la base de datos se representa en el nivel lógico de la misma forma que los datos ordinarios, de tal suerte que los usuarios autorizados puedan aplicar el mismo lenguaje relacional para consultarla, que aquél que emplean para con sus datos habituales.

5. Regla del sub-lenguaje de dato completo:

Se debe contar con un sub-lenguaje que contemple la definición de datos, la definición de vistas, la manipulación de datos, las restricciones de integridad, la autorización, el inicio y fin de una transacción.

6. Regla de actualización de vistas:

Todas las vistas que teóricamente sean actualizables deberán ser actualizadas por medio del sistema.

7. Inserción, actualización y eliminación de alto nivel:

La posibilidad de manejar una relación base o una relación derivada como un sólo operador se aplica a la lectura, inserción, modificación y eliminación de datos.

8. Independencia física de los datos:

Los programas de aplicación y la actividad en terminales no deberán ser afectados por cambios en el almacenamiento físico de los datos o en el método de acceso.

9. Independencia lógica de los datos:

Los programas de aplicación y la actividad en terminales no deberán ser afectados por cambios de cualquier tipo que preserven la información y que teóricamente permitan la no afectación en las tablas base.

10. Independencia de la integridad:

Las restricciones de integridad de una base de datos deberán poder definirse en el mismo sub-lenguaje de datos relacional y deberán almacenarse en el catálogo, no en los programas de aplicación.

11. Independencia de la distribución:

Un DBMS relacional tiene independencia de distribución.

12. Regla de la no subversión:

Si un sistema relacional tiene un lenguaje de bajo nivel (un sólo registro cada vez), ese bajo nivel no puede ser utilizado para suprimir las reglas de integridad y las restricciones expresadas en el lenguaje relacional de nivel superior (múltiples registros a la vez).

3.1.7. Lenguaje de consulta relacional

El lenguaje que se utiliza para organizar, administrar, y recuperar los datos almacenados en una base de datos relacional se le llama SQL o Structured Query Language (Lenguaje de consultas estructurado), este a su vez se apoya en el álgebra relacional que no es más que una serie de operaciones sobre relaciones (es decir utiliza álgebra relacional).

Entre las acciones que se pueden realizar con el lenguaje SQL son las siguientes:

- Creación y Eliminación de base de datos.
Instrucciones que se utilizan son: **create**, **alter** y **drop**.
- Creación, modificación y eliminación de tablas.
Instrucciones que se utilizan son: **create**, **alter** y **drop**.
- Inserción, Modificación, Consulta y Eliminación de datos
Instrucciones que se utilizan son: **select**, **insert**, **update** y **delete**.
- Creación de Vistas.
Instrucciones que se utilizan son: **create**, y **drop**.
- Definir privilegios.
Instrucciones: **grant** y **revoke**.

3.1.8. Modelo Entidad - Relación

Es una técnica de diseño de bases de datos relacionales en forma grafica, la cual nos permite representar los datos, información relativa a los mismos y sus relaciones, en este diseño se intenta plasmar la visión que se tiene sobre el mundo real dentro de un sistema informático [23][24][25].

Este modelo se caracteriza principalmente por:

- Sólo reflejar la existencia de los datos sin expresar lo que se hace con ellos.
- La independencia de la base de datos y de los sistemas operativos.
- La inclusión de todos los datos sin considerar las aplicaciones que se tendrán.

En este tipo de modelo se utilizan un conjunto de símbolos los cuales no ayudan a representar los datos como se muestra a continuación:

Entidades:

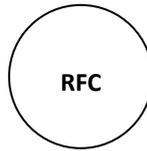
Se puede definir a una entidad como cualquier objeto real o abstracto, que existe en un contexto determinado. Y se representa por medio de un cuadrado, de la siguiente forma:



PROFESOR

Atributos:

A las características o propiedades de una entidad u objeto se le llaman atributos, y cada ejemplar de una misma entidad debe tener el mismo número de atributos.

**Claves:**

Cada entidad dentro del modelo entidad-relación debe contener un identificador, que puede ser un atributo o conjunto de atributos, los cuales identifican de manera única a cada ejemplar de una entidad, la forma de representar las claves es agregando dos líneas al atributo en la parte inferior del mismo.

Interrelaciones:

Las interrelaciones se refieren a las relaciones que existen entre las diferentes entidades que se tengan en un modelo, estas se representan por medio de líneas que unen a las diferentes entidades.

**3.1.9. Normalización**

La normalización es un proceso en el cual se utilizan una serie de restricciones, las cuales nos ayudan a eliminar posibles anomalías al momento de crear las entidades y las distintas relaciones entre las entidades.

El objetivo principal de la normalización es asegurar que el modelo conceptual que se genere funcione, y al momento de crear una base de datos no se tengan problemas al agregar, actualizar o borrar tuplas o registros.

Existen varias normas, pero las más importantes son tres y son las siguientes:

1ra. Forma Normal:

Todos los datos contenidos en los atributos de una relación (tabla) deben ser atómicos, esto quiere decir que solo debe existir un valor en la intersección de una fila con una columna. Los atributos de una entidad se deben dividir de tal forma que en cada atributo se represente una característica específica de la relación.

2da. Forma Normal:

Todos los atributos que no son clave primaria en una relación (tabla) dependen totalmente de su clave primaria.

3ra. Forma Normal:

El valor de un atributo no debe depender del valor de otro atributo de la misma tabla, esto quiere decir que no debe existir un arreglo de dependencias funcionales entre los atributos de una tabla.

3.2. Sistemas Gestores de Bases de Datos

El avance de la tecnología en el manejo de la información ayudo al desarrollo de nuevas herramientas para la administración de bases de datos, estos avances dieron lugar a los denominados sistemas gestores de base de datos (SGBD).

Los SGBD son sistemas que permiten mantener bases de datos, así como brindarnos las herramientas necesarias para el acceso, manejo, mantenimiento y seguridad de la información, permitiendo así a cualquier usuario crear y mantener sus propias bases de datos, liberándolos de transformar el nivel lógico en el que se diseñan al nivel físico que es la forma en que se guardan.

Los componentes funcionales que integran y controlan las partes específicas de un SGBD son:

Gestor de archivos:

Gestiona la asignación de espacio en la memoria del disco y de las estructuras de datos usadas para representar la información.

Manejador de bases de datos:

Sirve de interfaz entre los datos y los programas de aplicación.

Procesador de consultas:

Traduce las proposiciones en lenguajes de consulta a instrucciones de bajo nivel. Además convierte la solicitud del usuario en una forma más eficiente.

Compilador de DDL (Lenguaje de Definición de Datos).

Convierte las proposiciones DDL en un conjunto de tablas que contienen metadatos, estas se almacenan en el diccionario de datos.

Archivo de Datos:

En él se encuentran almacenados físicamente los datos de una organización.

Diccionario de datos:

Contiene la información referente a la estructura de la base de datos.

Índices:

Puede decirse que son listados que permiten un rápido acceso a registros específicos.

La siguiente imagen muestra los componentes funcionales de las base de datos:

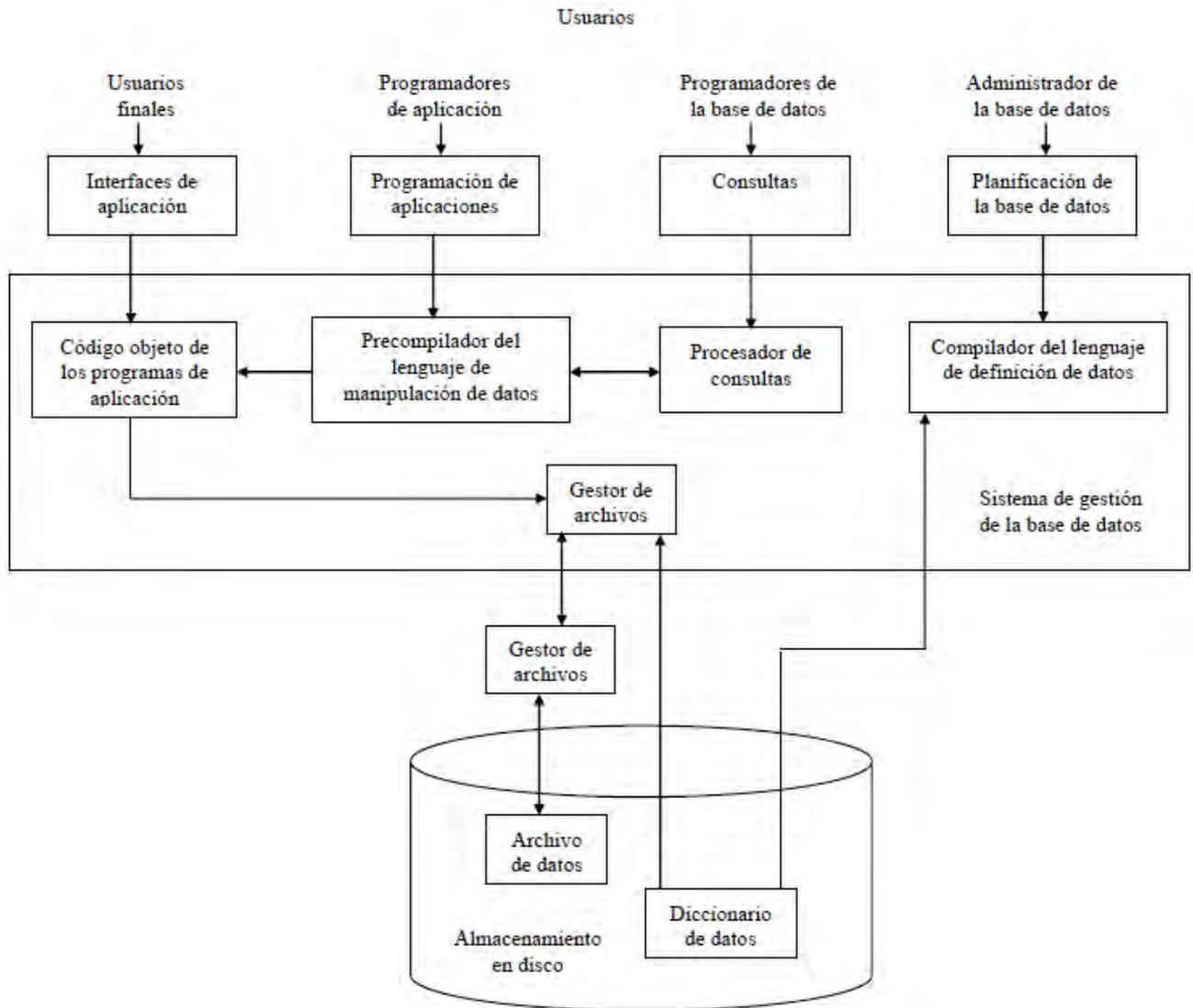


Figura 3.4 Componentes funcionales de las base de datos.

3.2.1. Clasificación de los Sistemas Gestores de Bases de Datos

Existen muchas formas de clasificar a los manejadores de bases de datos, a continuación se muestran algunas clasificaciones más importantes.

- **Clasificación por ámbito:**

En esta clasificación tenemos dos subconjuntos generales que son:

- **Los comerciales** (SQL Server, Sybase, Oracle, Informix, DB2), que se manejan por el tipo de licencia:
 - Licencia por usuarios conectados.
 - Licencia por procesador.
- **Los Software libre** (PostgreSQL, Mysql), que no cobran por el uso del producto.

- **Clasificación por el volumen de información que manejan:**

- **Corporativos** (Oracle, Informix, Sybase, DB2, PostgreSQL).
- **Departamental** (SQL Server, SQL Anywhere, Mysql).

Capítulo 4 Análisis y diseño del Sistema de Asignaciones Docentes de la Facultad de Estudios Superiores Cuautitlán – Módulo de UAPA

4.1. Análisis y requerimientos básicos del usuario

La fase de análisis es una de las etapas más importantes en el desarrollo de un sistema, como se puede observar en los diferentes modelos de ciclo de vida del capítulo 2, ya que el resultado que entregue el análisis será la base para el diseño e implementación del sistema; El desarrollo erróneo del mismo involucra no cumplir con los requerimientos planteados por el usuario, así como la generación de costos en tiempo y dinero para resarcir los errores en las etapas siguientes (como el diseño e implementación del sistema). Por lo anterior, se llevaron a cabo entrevistas con los diferentes actores del sistema, así como un estudio minucioso de la estructura organizacional de la FESC y los procesos que llevan a cabo cada una de las áreas involucradas con el desarrollo de las actividades académico-administrativas, también se recabo la documentación generada en cada uno de los procesos y se estudio el estatuto del personal académico EPA para delimitar las políticas o restricciones que se manejan al momento de realizar la contratación de un profesor.

Para el desarrollo de esta tesis se analizó específicamente el módulo de la Unidad de Asuntos del Personal Académico (UAPA), a continuación se describen las actividades académico-administrativas que lleva a cabo dicha área para el levantamiento de sus requerimientos:

La UAPA es el área encargada de revisar y analizar la documentación necesaria para la contratación de los académicos, esta documentación es entregada por los Departamentos Académicos, y va desde la currícula del profesor hasta la hoja de asignación⁸; después de que la UAPA lleva a cabo la revisión de la documentación y verificar si cumple con los requerimientos establecidos en el EPA, de acuerdo con el tipo de nombramiento que tenga asignado el académico (ver tabla 4.1) se establecen dos casos:

- 1) Si la documentación no cumple con los requerimientos establecidos en el EPA: UAPA devuelve la información al Departamento Académico correspondiente para que realice las correcciones necesarias.
- 2) Si la documentación es correcta, UAPA arma paquetes por departamento académico con la documentación de los docentes, para que en una de las sesiones de Consejo Técnico se apruebe la asignación de actividades académicas a cada uno de los mismos.

⁸ Hoja de asignación: Documento interno de la FESC donde se muestra el o los nombramientos de un académico, así como las actividades de docencia directa y de apoyo académico que tiene asignadas en un semestre determinado

Nombramiento	Categoría	Nivel	Tiempo	Situación Contractual
Ayudante de Profesor		A	Horas	
Ayudante de Profesor		B	Horas	
Profesor de Asignatura		A	Horas	Definitivo
Profesor de Asignatura		A	Horas	Interino
Profesor de Asignatura		B	Horas	Definitivo
Profesor de Asignatura		B	Horas	Interino
Profesor de Carrera	Asociado	A	Medio Tiempo	Definitivo
Profesor de Carrera	Asociado	A	Medio Tiempo	Interino
Profesor de Carrera	Asociado	A	Tiempo Completo	Definitivo
Profesor de Carrera	Asociado	A	Tiempo Completo	Interino
Profesor de Carrera	Asociado	B	Medio Tiempo	Definitivo
Profesor de Carrera	Asociado	B	Medio Tiempo	Interino
Profesor de Carrera	Asociado	B	Tiempo Completo	Definitivo
Profesor de Carrera	Asociado	B	Tiempo Completo	Interino
Profesor de Carrera	Asociado	C	Medio Tiempo	Definitivo
Profesor de Carrera	Asociado	C	Medio Tiempo	Interino
Profesor de Carrera	Asociado	C	Tiempo Completo	Definitivo
Profesor de Carrera	Asociado	C	Tiempo Completo	Interino
Profesor de Carrera	Titular	A	Medio Tiempo	Definitivo
Profesor de Carrera	Titular	A	Medio Tiempo	Interino
Profesor de Carrera	Titular	A	Tiempo Completo	Definitivo
Profesor de Carrera	Titular	A	Tiempo Completo	Interino
Profesor de Carrera	Titular	B	Medio Tiempo	Definitivo
Profesor de Carrera	Titular	B	Medio Tiempo	Interino
Profesor de Carrera	Titular	B	Tiempo Completo	Definitivo
Profesor de Carrera	Titular	B	Tiempo Completo	Interino
Profesor de Carrera	Titular	C	Medio Tiempo	Definitivo
Profesor de Carrera	Titular	C	Medio Tiempo	Interino
Profesor de Carrera	Titular	C	Tiempo Completo	Definitivo
Profesor de Carrera	Titular	C	Tiempo Completo	Interino
Técnico Académico	Asociado	A	Medio Tiempo	Definitivo
Técnico Académico	Asociado	A	Medio Tiempo	Interino
Técnico Académico	Asociado	A	Tiempo Completo	Definitivo
Técnico Académico	Asociado	A	Tiempo Completo	Interino
Técnico Académico	Asociado	B	Medio Tiempo	Definitivo
Técnico Académico	Asociado	B	Medio Tiempo	Interino
Técnico Académico	Asociado	B	Tiempo Completo	Definitivo
Técnico Académico	Asociado	B	Tiempo Completo	Interino
Técnico Académico	Asociado	C	Medio Tiempo	Definitivo
Técnico Académico	Asociado	C	Medio Tiempo	Interino

Técnico Académico	Asociado	C	Tiempo Completo	Definitivo
Técnico Académico	Asociado	C	Tiempo Completo	Interino
Técnico Académico	Auxiliar	A	Medio Tiempo	Definitivo
Técnico Académico	Auxiliar	A	Medio Tiempo	Interino
Técnico Académico	Auxiliar	A	Tiempo Completo	Definitivo
Técnico Académico	Auxiliar	A	Tiempo Completo	Interino
Técnico Académico	Auxiliar	B	Medio Tiempo	Definitivo
Técnico Académico	Auxiliar	B	Medio Tiempo	Interino
Técnico Académico	Auxiliar	B	Tiempo Completo	Definitivo
Técnico Académico	Auxiliar	B	Tiempo Completo	Interino
Técnico Académico	Auxiliar	C	Medio Tiempo	Definitivo
Técnico Académico	Auxiliar	C	Medio Tiempo	Interino
Técnico Académico	Auxiliar	C	Tiempo Completo	Definitivo
Técnico Académico	Auxiliar	C	Tiempo Completo	Interino
Técnico Académico	Titular	A	Medio Tiempo	Definitivo
Técnico Académico	Titular	A	Medio Tiempo	Interino
Técnico Académico	Titular	A	Tiempo Completo	Definitivo
Técnico Académico	Titular	A	Tiempo Completo	Interino
Técnico Académico	Titular	B	Medio Tiempo	Definitivo
Técnico Académico	Titular	B	Medio Tiempo	Interino
Técnico Académico	Titular	B	Tiempo Completo	Definitivo
Técnico Académico	Titular	B	Tiempo Completo	Interino
Técnico Académico	Titular	C	Medio Tiempo	Definitivo
Técnico Académico	Titular	C	Medio Tiempo	Interino
Técnico Académico	Titular	C	Tiempo Completo	Definitivo
Técnico Académico	Titular	C	Tiempo Completo	Interino
Investigador	Asociado	C	Tiempo Completo	Definitivo
Investigador	Asociado	C	Tiempo Completo	Interino
Investigador	Titular	B	Tiempo Completo	Definitivo
Investigador	Titular	B	Tiempo Completo	Interino
Investigador	Titular	C	Tiempo Completo	Definitivo
Investigador	Titular	C	Tiempo Completo	Interino

Tabla 4.1 Nombramientos de la Facultad de Estudios Superiores Cuautitlán.

Requerimientos básicos del usuario UAPA

Una vez realizado el análisis de requerimientos a la UAPA se determinaron las siguientes características que debe cubrir el sistema para que el área pueda llevar a cabo sus actividades dentro del nuevo sistema.

Requiere que el sistema:

- Le permita ver los nombramientos del profesor por departamento y sección académica.
- Le permita visualizar la hoja de asignación del académico.
- Evalué que el o los nombramientos del profesor cumplan con los mínimos y máximos de docencia directa y de actividades de apoyo establecidos en el EPA.
- Evalué que las actividades del profesor no excedan de 8 horas diarias.
- Permita enviar a Consejo Técnico los nombramientos que ya fueron revisados y cumplen con normatividad.
- Le permita generar:
 - Reporte de profesores que ya han sido revisados y enviados a Consejo Técnico para su aprobación en una sesión.
 - Reporte de profesores que se rechazó su asignación de actividades académicas.

Una vez establecidos los requerimientos del usuario, se realizó la documentación de los mismos, para esta tarea se utilizó una herramienta CASE (las cuales se mencionaron en el capítulo 2) llamada UML, (Unified Modeling Language - Lenguaje Unificado de Modelado), esta herramienta utiliza un lenguaje gráfico para construir, documentar, visualizar y especificar un sistema de software. Con esta información se generó un módulo para cubrir las necesidades específicas de la UAPA.

4.2. Características del sistema

4.2.1. Hardware y Software

Otro de los aspectos más importantes a la hora de desarrollar un sistema es determinar las características de hardware y software que soportaran la aplicación, para este sistema se requirió un servidor con las siguientes características:

- Servidor Dell PowerEdge R510 con 15gb de memoria RAM, 8 procesadores Intel Xeon a una velocidad de 2.67GHz y disco duro de 1Tb

Dada la robustez del servidor permitirá no sólo hospedar esta aplicación sino las siguientes que se tenga planeadas desarrollar para complementar el sistema actual, por ello es importante dar holgura en cuanto a hardware, asimismo estas características van aunadas a los requerimientos que se tienen respecto al software:

Para el desarrollo de esta aplicación se instaló en siguiente software:

Sistema operativo:

Un Sistema operativo (SO) es un software que actúa de interfaz entre los dispositivos de hardware y los programas usados por el usuario para utilizar un equipo de cómputo [26].

El sistema operativo instalado es **Fedora 13**, está basado en Linux que ofrece software libre y de código abierto, es construido por personas de todo el planeta que trabajan juntas como una comunidad, por tanto no es necesario comprar una licencia para poder utilizarlo.

Requerimientos de procesador y memoria:

Procesador:

- Para modo texto: Pentium Pro 200 MHz o superior.
- Para gráficos: Pentium Pro 400 MHz o superior.

Memoria RAM

- Mínimo para modo texto: 256 MB.
- Mínimo para gráficos: 384 MB.
- Recomendado para gráficos: 512 MB.

Disco duro:

Todos los paquetes pueden llegar a ocupar en total hasta 9 GB de espacio en disco. El tamaño final está determinado por los paquetes que se hayan seleccionado.

Servidor de aplicaciones:

En informática, se denomina **servidor de aplicaciones** a un servidor en una red de computadores que ejecuta ciertas aplicaciones. Usualmente se trata de un software que proporciona servicios de aplicación a las computadoras cliente. Un servidor de aplicaciones generalmente gestiona la mayor parte o la totalidad de las funciones de lógica de negocio y de acceso a los datos de la aplicación [27].

Para este proyecto se utilizó el servidor de aplicaciones **GlassFish Server 3.0.1** de software libre desarrollado por Sun Microsystems el cual tiene los siguientes requerimientos:

Memoria RAM

- Mínimo: 512 MB.

Disco duro:

- Mínimo: 250 MB.

Software:

- Mínimo Linux Fedora 10.
- Tener instalado java JDK 6 versión 1.6.0 mínimo.

Java:

Es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90. Actualmente las aplicaciones desarrolladas en java se ejecutan en más de tres mil millones de dispositivos, entre los que se incluyen teléfonos móviles, lectores de libros electrónicos, teléfonos VOIP, televisores y decodificadores, reproductores de Blu-ray, routers, impresoras, etc. [28][29]

Requerimientos:

Memoria RAM

- Mínimo Linux: 64 MB.
- Mínimo Windows: 128 MB.

Disco duro:

- 58 MB.

Servidor de base de datos:

Como se menciona en el capítulo 3 los **sistemas gestores de bases de datos** (en inglés Database Management System, abreviado *DBMS*) son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Para administrar la información del sistema SAFESC se utilizó PostgreSQL versión 9.0.1, ya que es uno de los manejadores de bases de datos objeto-relacionales más potentes, además de ser de código libre, por lo tanto no es necesario adquirir una licencia para utilizarlo, permitiendo reducir costos.

Disco duro:

- Depende de la cantidad de información a manejar

Sistema operativo:

- Es multiplataforma, por lo que puede instarse en cualquier sistema operativo sea Windows o cualquier versión de Linux

Cliente:

Como se menciona en el capítulo 2, el cliente es el proceso que permite al usuario formular una petición y pasarla al servidor para recibir una respuesta.

Para esta aplicación se requiere del lado del cliente tener instalado **Mozilla Firefox** que es un navegador web libre y de código abierto desarrollado por la Fundación Mozilla, es el segundo navegador más utilizado de Internet con más de 450 millones de usuarios.

Como se puede observar el software instalado cuenta con la robustez necesaria para soportar la aplicación SAFESC, además de contar con documentación y actualizaciones en línea que permiten mantener a cada uno de los componentes al día, lo cual ofrece mayor seguridad y confiabilidad al sistema, aunado a esto se utiliza Struts de Java que brinda un mayor control sobre la aplicación como se menciona a continuación:

4.2.2. Struts

Struts es una herramienta de soporte para el desarrollo de aplicaciones Web en la plataforma Java EE (Java Enterprise Edition). Struts se desarrollaba como parte del proyecto Jakarta de la Apache Software Foundation, pero actualmente es un proyecto independiente conocido como Apache Struts y se basa en el patrón de arquitectura de software Modelo-Vista-Controlador (MVC).

Modelo Vista Controlador (MVC) es un patrón de diseño de arquitectura de software que estructura como debe ser desarrollada una aplicación, así como las capas que van a componer la misma y la funcionalidad de cada una.

Según este patrón la capa intermedia de una aplicación web puede ser dividida en tres bloques funcionales y son: Modelo, Vista, Controlador.

4.2.2.1. Modelo

En la arquitectura MVC la lógica de negocio de la aplicación, así como el acceso a los datos y su manipulación se encuentran encapsulados dentro del modelo, el Modelo son todas las clases .java que el desarrollador tiene que implementar.

4.2.2.2. Vista

Es la encargada de generar las respuestas que deben ser enviadas al cliente, normalmente esta respuesta debe incluir datos proporcionados por el Controlador, por lo tanto su implementación en java por parte del desarrollador correrá a cargo de una página **JSP**.

4.2.2.3. Controlador

Se puede decir que el Controlador es el cerebro de toda la aplicación, todas las peticiones que realice el cliente hacia la capa intermedia son dirigidas al controlador, el cual tiene como objetivo determinar las acciones a realizar para cada una de las peticiones e invocar a cada uno de los componentes de la aplicación (Modelo y Vista) para que realicen las acciones requeridas en cada caso, encargándose así de la coordinación de todo el proceso.

La API⁹ de Struts está formada por conjunto de clases de apoyo que permiten estructurar las aplicaciones y ayudan al Controlador a realizar su función, a continuación se mencionan algunos de los elementos más importantes:

ActionServlet

El Controlador se puede definir por la clase Action servlet, y constituye el punto de entrada de la aplicación, recibiendo todas las peticiones HTML que llegan del cliente, a fin de determinar la operación a realizar, esta función la desarrolla al extraer la última parte de la URL y la contrasta con la información contenida en el archivo de configuración llamado **struts-config.xml**, cada acción va asociada a un **ActionForm**, el cual será rellenado con los datos procedentes del cliente y pasa el control a un objeto **Accion** encargado de procesarla.

Action: Son los objetos responsables de procesar los distintos tipos de peticiones que llegan a la aplicación (ActionServlet), el principal método con el que cuenta es **execute()**, por lo tanto el programador deberá definir una subclase de Action y tendrá que sobrescribir el método execute(), incluyendo en él las instrucciones necesarias para el tratamiento de la

⁹ API (Interfaz de programación de aplicaciones): Es un conjunto de funciones y procedimientos que ofrecen ciertas bibliotecas y representan un método para conseguir abstracción en la programación. Uno de los principales propósitos de una API consiste en proporcionar un conjunto de funciones de uso general, por ejemplo, para dibujar ventanas o iconos en la pantalla.

petición como llamadas a los métodos de la lógica de negocio implementada en el Modelo o la transferencia de resultados a las Vistas para su presentación.

ActionForm: Los objetos ActionForm son los encargados de facilitar el transporte de los datos entre las diferentes capas de la aplicación, son utilizados por los ActionServlet para capturar los datos procedentes de los formularios y enviárselos al objeto Action correspondiente, el programador deberá proporcionar los datos necesarios para su almacenamiento, así como crear los métodos set/get que den acceso a los mismos.

ActionMapping: Un objeto de tipo ActionMapping representa la asociación entre la petición y el objeto Action que la tiene que procesar, esto quiere decir que contiene la información sobre el path o URL que provoca la ejecución de la acción, así como las posibles Vistas que se pueden presentar al cliente tras su procesamiento.

Cuando el Controlador ActionServlet invoca al método execute() de un objeto Action para procesar una petición, proporciona como parámetro un objeto ActionMapping, de esta manera el programador puede dirigir al cliente a cualquiera de las Vistas asociadas al objeto una vez que ha sido procesada la petición

ActionForward: Un objeto de la clase ActionForward permite trabajar con direcciones virtuales (dar un alias a un archivo), además de permitir transferir una petición desde código o redireccionar al usuario a una página determinada.

Ejemplo de funcionamiento del modelo MVC:

Supongamos que al momento de recibir una petición del Cliente, se requiere enviar como respuesta determinada información existente en una base de datos, entonces el Controlador recibe la petición y solicitará los datos necesarios al modelo, una vez recibidos, se los proporcionará a la Vista para que ésta les aplique el formato de presentación correspondiente y envíe la respuesta al cliente figura 4.2.

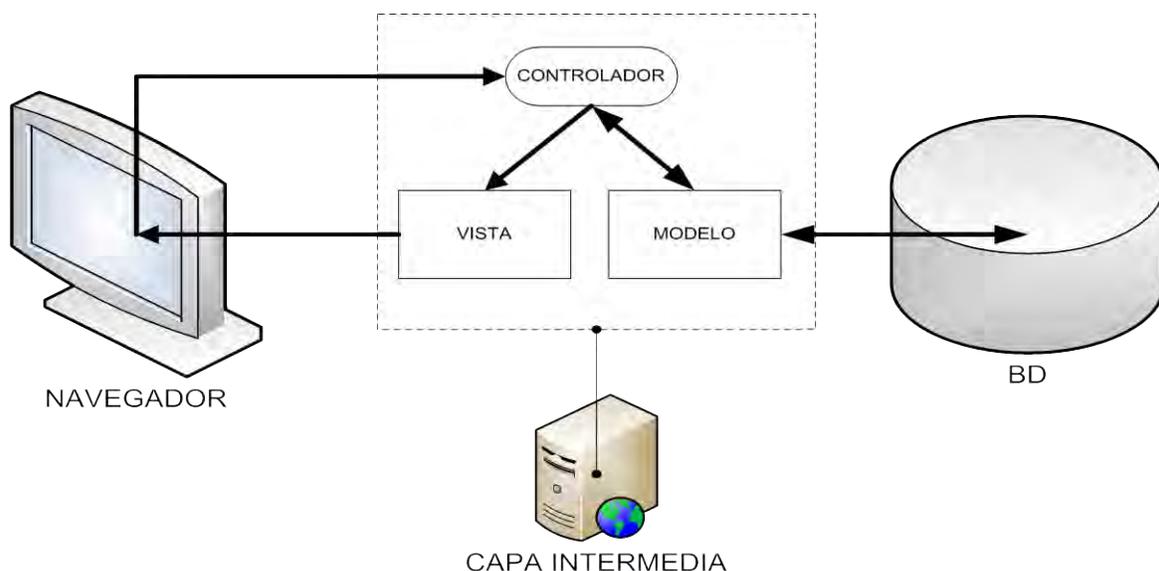


Figura 4.2 Diagrama de flujo del modelo MVC.

4.2.3. Niveles de seguridad

Para garantizar la integridad de la información que se maneja en este sistema se tiene distintos niveles de seguridad los cuales se mencionan a continuación:

a) Sistema operativo

A nivel sistema operativo se establecieron los siguientes controles:

- Control de acceso mediante usuarios para garantizar que solo el personal autorizado pueda entrar al mismo de manera local o remota.
- Se configuro el firewall para solo permitir conexiones a los puertos necesarios para el funcionamiento del sistema.

b) Aplicación

A nivel aplicación se crearon distintos usuarios que de acuerdo a su función dentro del sistema se les muestra un menú con opciones diferentes. Así mismo se cifro su nombre de usuario y contraseña mediante el algoritmo de encriptación SHA1, lo cual permitirá mantener un nivel de seguridad alto y garantizará que ningún usuario que tenga acceso a la base de datos pueda conocer los nombres de usuario y contraseña. Los usuarios generados para la aplicación son:

- Departamentos académicos
- Jefes de Sección académica
- Coordinaciones de carrera
- Consejo Técnico
- UAPA
- Administrador
- Personal
- Consulta

c) Base de datos

Para poder acceder a la base de datos se crearon distintos tipos de usuarios a los cuales se les asignaron diferentes tipos de permisos de acuerdo a las funciones que desarrollan (respaldos, actualizaciones, vistas), además de configurarse políticas de acceso para que la base de datos solo pueda ser accedida de manera local o mediante la aplicación.

4.3. Diseño de la base de datos

Como se ha mencionado en capítulos anteriores, uno de los aspectos importantes a cubrir al momento de desarrollar un sistema es el desarrollar un diseño de la base de datos que contendrá la información, ya que planear cómo se almacenará, actualizará y eliminará la información permitirá el buen manejo de la misma, en caso contrario no se tendrá ningún control de la forma en cómo se procesa la información y como trabajará el sistema, a continuación se muestra el diagrama datos del sistema SAFESC y el diccionario de datos de la base de la base de datos se muestra en el apéndice A

4.4.Descripción del módulo de la Unidad de Asuntos del Personal Académico (UAPA)

Como ya se explico en el primer capítulo, UAPA es el área encargada de revisar los requisitos académicos para la contratación de los académicos, a continuación se da una explicación detallada de las tareas primordiales que se desarrollan en el módulo de UAPA y se explica de forma detallada cómo trabaja el modelo MVC.

Como cualquier usuario del sistema el primer paso a realizar es identificarse, para poder acceder al mismo:

The image shows a web browser window displaying the login page of the Universidad Nacional Autónoma de México (UNAM). The header includes the UNAM logo and the text 'Universidad Nacional Autónoma de México'. Below the header is a navigation bar with links for 'Mapa de sitio', 'Preguntas frecuentes', and 'Contacto'. The main content area features a large '36' logo with the word 'UNIVERSIDAD' underneath. Below the logo are two input fields: 'USUARIO:' and 'CLAVE:'. At the bottom of the form is a button labeled 'Ingresar'.

Figura 4.4 Página para insertar nombre de usuario y contraseña.

Una vez autenticado el usuario UAPA aparecerá una Vista donde deberá seleccionar el ciclo escolar sobre el cual desea trabajar, asimismo con el fin de explicar el funcionamiento del modelo MVC utilizaremos este proceso:

Al seleccionar el ciclo escolar y dar clic en aceptar en la página `selsemestre.jsp` estamos realizando una petición al servidor ver figura 4.5.



The screenshot shows the UAPA2 web application interface. At the top, there is a header for Universidad Nacional Autónoma de México (UNAM) with the logo and the text "UNAM Universidad Nacional Autónoma de México". Below this is a navigation bar with links for "Mapa de sitio", "Preguntas frecuentes", and "Contacto". The main content area is titled "UAPA" and features a sidebar with "REVISIÓN" and "REPORTES" options. The main content displays "Bienvenido UAPA2" and a message: "El semestre 2012-1 se encuentra disponible sólo para consulta y reportes". Below this message is a form with a "Semestre" label, a dropdown menu showing "2012-1", and an "Aceptar" button. A "Salir del sistema" link is located at the bottom of the form area.

Figura 4.5 `selsemestre.jsp`, página para seleccionar el ciclo escolar.

El código de `selsemestre.jsp` (ver figura 4.6) hace uso del Form `SelsemestreForm` y de sus propiedades **`idusuario`**, **`pagorigen`** y **`uciclo`** para almacenar la información del formulario y pasarla al Modelo, la propiedad **`idusuario`** almacenará el identificador del usuario que está realizando la petición, **`pagorigen`** contendrá el nombre de la página donde se invoca el Form y **`uciclo`** almacenará el ciclo escolar seleccionado por el usuario.

Para poder utilizar las propiedades de `SelsemestreForm` previamente se debió haber creado con sus métodos `set/get`, los cuales permitirán tener acceso a los datos, ver figura 4.8.

```

<html:form action="selsemestre">
  <html:hidden name="SelsemestreForm" property="idusuario" value='<%= idu.toString() %>' />
  <html:hidden name="SelsemestreForm" property="pagorigen" value='<%= pagorigen.toString() %>' />
  <table width="730" border="1">
    <tr>
      <th scope="col">Semestre</th>
    </tr>
    <tr>
      <th scope="col">
        <label>
          <html:select property="uciclo1" name="SelsemestreForm">
            <html:option value="2">2012-1</html:option>
            <html:option value="3">2012-2</html:option>
            <html:option value="4">2013-1</html:option>
            <html:option value="5">2013-2</html:option>
          </html:select>
        </label></th>
    </tr>
    <tr>
      <td colspan="5">
        <label>
          <div align="center">
            <html:submit value="Aceptar"/>
          </div>
        </label>
      </td>
    </tr>
  </table>
</html:form>

```

Figura 4.6 selsemestre.jsp.

```

public class SelsemestreForm extends ActionForm{

    private String uciclo1;
    private String pagorigen;
    private String idusuario;

    public String getuciclo1(){
        return uciclo1;
    }
    public String getpagorigen(){
        return pagorigen;
    }
    public String getidusuario(){
        return idusuario;
    }
    public void setuciclo1(String puciclo1){
        uciclo1 = puciclo1.trim();
    }
    public void setpagorigen(String pagorigen1){
        pagorigen = pagorigen1.trim();
    }
    public void setidusuario(String pidusuario){
        idusuario = pidusuario.trim();
    }
    public SelsemestreForm(){
        super();
    }
}

```

Figura 4.7 SelsemestreForm.jsp.

En el código de la Figura 4.7 se declara el Form SelsemestreForm junto con sus propiedades idusuario, pagorigen y uciclo con sus respectivos métodos get/set.

Una vez realizada la petición por parte del Cliente (Vista), la recibe el Controlador mediante una clase llamada ActionServlet (la cual se encuentra declarada en el archivo web.xml ver figura 4.8), la interpreta y de acuerdo con la configuración del archivo struts-config.xml (ver figura 4.9) determina la operación a realizar, asocia un Form (en este caso SelsemestreForm) y dirige la petición mediante el objeto Action a un proceso determinado que se encuentran expuesto por el Modelo (Selsemestre.java)

```
<servlet>
  <servlet-name>action</servlet-name>
  <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
  <init-param>
    <param-name>config</param-name>
    <param-value>/WEB-INF/struts-config.xml</param-value>
  </init-param>
</servlet>
```

Figura 4.8 ActionServlet.

```
<struts-config>
  <form-beans>
    <form-bean name="SemestreForm" type="formBean.SemestreForm" />
  </form-beans>
  <action-mappings>
    <action input="/selsemestre.jsp" name="SelsemestreForm" path="/selsemestre" type="actionBean.Selsemestre" >
      <forward name="success" path="/selsemestre.jsp"/>
      <forward name="failure" path="/errorSelSem.html"/>
    </action>
  </action-mappings>
</struts-config>
```

Figura 4.9 Struts-config.xml.

Como ya se menciono el archivo Struts-config.xml nos sirve para configurar las acciones que llevará a cabo la clase ActionServlet, las dos configuraciones a realizar en este archivo son las siguientes:

- a) Declarar todos los Form que utilicemos, tal es el caso de SelsemestreForm, como se puede observar entre las etiquetas **<form-beans>** (ver figura 4.9), para ello se tiene el atributo **name**, que es el nombre con el que se identificara el Form y **type**, que guarda el paquete donde se encuentra el Form y su nombre.
- b) Dentro de las etiquetas **<action-mappings>** se registran todas las clases Action de una aplicación y utilizando un subelemento llamado **<action>** para definir los parámetros asociados a cada acción, a continuación se describen los atributos indicados en el elemento action:
 - **Path:** Representa la dirección que debe ser utilizada en la URL de la petición para que la clase ActionServlet envíe la petición a un objeto action específico.
 - **Name:** Nombre del ActionForm que será utilizado para guardar los datos del formulario cliente.
 - **Type:** Es el nombre de la subclase action que llevará a cabo las operaciones en este caso selSemeste.java (Modelo)
 - **Input:** Indica la URL de la Vista que debe ser enviada al cliente en caso que se genere algún error durante alguna validación.

Como se puede observar también existen elementos **<forward>** (ver figura 4.9), los cuales representan las posibles Vistas a las que se puede encaminar al cliente como respuesta del action, en este caso solo tenemos dos Vistas: una cuando el Modelo devuelve un valor “success” y otra cuando devuelve “failure”.

En nuestro ejemplo, una vez que **<action-mapping>** hace el llamado al Modelo (selSemestre.java), este recibe los valores que tiene cargados el SelsemestreForm y se encarga de verificar las fechas para las cuales el sistema está habilitado para que trabaje el usuario UAPA, si se encuentra dentro de un periodo valido, Selsemestre.java devolverá un ActionForward con el valor: “success” permitiendo al usuario el acceso a las demás opciones del sistema, sino devolverá un ActionForward con el valor: “failure” que no permitirá al usuario utilizar las opciones del sistema, regresando al usuario a la Vista necesaria para cada caso selsemestre.jsp o errorSelSem.html respectivamente, ver figura 4.10.

```

SelsemestreForm lf = (SelsemestreForm)form;
HttpSession hs=request.getSession();
hs.setAttribute("idcicloescolar",lf.getuciclo1());
querys query = new querys();
String qry = "Select * from apertura where idcicloescolar = "+lf.getuciclo1()+" and
idmodulo="+hs.getAttribute("idmodulo")+ " and (usuario="+lf.getidusuario()+
" or usuario=500)";
System.out.println("usuario segun ivan: " +qry);

try{
    ResultSet rs = query.consultar(qry);
    Integer tempo=0;
    while (rs.next()) {
        calendario calfi1 = new calendario(rs.getString("fechaapertura"));
        calendario calft1 = new calendario(rs.getString("fechacierre"));
        Date now = Calendar.getInstance().getTime();
        if( now.after(calfi1.getdate()) && now.before(calft1.getdate())){
            tempo=tempo+1;
        } else {
            tempo=tempo;
        }
    }
    if (tempo!=0){
        hs.setAttribute("mopen",1);
        response.sendRedirect(lf.getpagorigen());
    }else{
        hs.setAttribute("mopen",0);
        response.sendRedirect(lf.getpagorigen());
    }
} catch(Exception e) {
    return mapping.findForward("failure");
}

```

Figura 4.10 Selsemestre.java.

Una vez establecido el semestre en el que se desea trabajar, el usuario puede seleccionar el departamento y sección académica, ver figura 4.11 y 4.12



Figura 4.11 seldepto.jsp, para seleccionar un departamento académico.

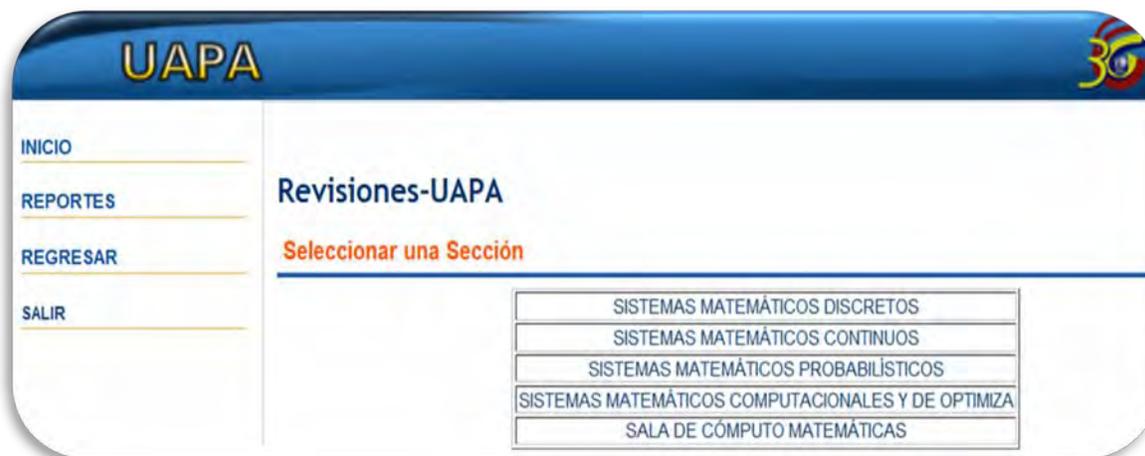


Figura 4.12 selseccion.jsp, para seleccionar una sección académica.

Una vez que el usuario ha seleccionado el departamento y sección académica, se muestran todos los nombramientos que han sido enviados a revisión por dicha sección, previamente el sistema realizó un primer filtro de los nombramientos enviados a UAPA para comprobar si cumple con las especificaciones del Estatuto del Personal Académico, como se muestra en la figura 4.13

No.	Profesor	Nombramiento	Hrs	Mov	Status Revisión
1	CARMONA CARRASCO ARACELI	Prof Asig A, i	6.0	REINGRESO	NO Revisado
	DDCV/HMM/082/01/2012	Del 2012-01-30	Al 2012-08-05	Diseño y Comunicación Visual	DISEÑO DE ILUSTRACIÓN + SIMBOLOGÍA Y DIS SOP. TRIDI
2	ZAPATA FLORES CARMEN	Prof Asig A, i	6.0	OTRO NOMBRAMIENTO	NO Revisado
	DDCV/HMM/083/01/2012	Del 2012-01-30	Al 2012-08-05	Diseño y Comunicación Visual	DISEÑO DE ILUSTRACIÓN + SIMBOLOGÍA Y DIS SOP. TRIDI
3	RANGEL RODRIGUEZ GLORIA VIRGINIA	Prof Asig A, i	11.0	OTRO NOMBRAMIENTO	NO Revisado
	DDCV/HMM/068/01/2012	Del 2012-01-30	Al 2012-08-05	Diseño y Comunicación Visual	DISEÑO DE ILUSTRACIÓN + SIMBOLOGÍA Y DIS SOP. TRIDI
4	ROBLEDO RAMÍREZ ERIKA	Prof Asig A, i	6.0	NUEVO INGRESO	NO Revisado
	DDCV/HMM/095/02/2012	Del 2012-02-07	Al 2012-08-05	Diseño y Comunicación Visual	DISEÑO DE ILUSTRACIÓN + SIMBOLOGÍA Y DIS SOP. TRIDI

Figura 4.13 revision-uapa.jsp, para seleccionar un nombramiento en específico.

Para poder mostrar la vista de la figura 4.13 es necesario realizar una consultas a la base de datos que nos permitan visualizar dicha información, como se puede mostrar en el código de la figura 4.14.

```

data datdepp2 = new data("profesor_depto_secc","departamento_iddepartamento",idep);
ResultSet rsdepp2 = datdepp2.getrs();
String profs2="";
try{
while(rsdepp2.next()){
profs2=profs2+rsdepp2.getString("profesor_idprofesor")+",";
}
profs2=profs2.substring(0,profs2.length()-1);
consuenta = "select count(*) from oficio_asignacion where cicloescolar_idcicloescolar="+ciclo+ ""
+ " and status="+statustr+"" and usuario_idusuario in (select idusuario from useccion where "
+ " idseccion in (select idseccion from seccion where iddepartamento =" +idep +") "
+ "and profesor_idprofesor in("+profs2+"");
}
catch(Exception e){
System.out.print(e);
}

```

Figura 4.14 revision-uapa.jsp, para seleccionar un nombramiento en específico.

Para poder realizar la consulta a la base de datos primero fue necesario crear un objeto de la clase data (clase que permite realizar una conexión a la base de datos) con el nombre datdepp2, además de un ResultSet que nos permitirá almacenar el resultado de la consulta para su posterior despliegue en pantalla, ver figura 4.14.

Una vez que se muestran los nombramientos de los académicos, el usuario UAPA puede seleccionar un nombramiento en específico para revisar de manera más completa las asignaciones de docencia directa y actividades de apoyo y poder determinar si las asignaciones son correctas o no, para lo cual se tienen dos resultados:

- a) Si las actividades de docencia y actividades de apoyo son correctas, de acuerdo con el Estatuto del Personal Académico, el nombramiento es enviado a Consejo Técnico para su aprobación.
- b) Si es incorrecta la asignación de actividades, el nombramiento es rechazado para que la sección correspondiente realice los cambios necesarios y vuelva a enviar el nombramiento a UAPA con su propuesta de asignaciones.

En la figura 4.15 se muestran las actividades de docencia y de apoyo asignadas a un nombramiento para que UAPA rechace o apruebe la propuesta dando clic en “verifica” o “rechaza” respectivamente.

Profesor CARMONA CARRASCO ARACELI
SEMESTRE 2012-2
Prof Asig A, i con 6.0 HRS

Svo. Semestre

Clave	Asignatura	Cr	Grupo	Aula	LUNES	MARTES	MIERCOLES	JUEVES	VIERNES	SÁBADO	HT
855	ILUSTRACION II (10222)	8	2852	A2-202	-	-	-	-	14:00-16:00	-	2.0
855	ILUSTRACIÓN II (10222)	8	2852A	A2-202	-	-	-	-	16:00-20:00	-	4.0
HORAS TOTALES DE DOCENCIA											6.0

[Actividades de Apoyo - Tabla de Equivalencias](#)

Tipo	Descripción	Ubicación	LUNES	MARTES	MIERCOLES	JUEVES	VIERNES	SÁBADO	HT	
HORAS TOTALES DE APOYO										0.0

Actividades Intersemestrales

Tipo	Descripción	Ubicación	LUNES	MARTES	MIERCOLES	JUEVES	VIERNES	SÁBADO	HT	
AA-A	ELABORACIÓN DE MATERIAL DIDACTICO (APUNTES) PARA LA MATERIA DE ILUSTRACIÓN II.	A202	-	-	-	-	14:00-20:00	-	6.0	
HORAS TOTALES DE INTERSEMESTRE										6.0

[Verifica](#) [Rechaza](#) [Regresar](#) [Hoja de Asignacion](#)

Figura 4.15 nombramiento-uapa.jsp, para aprobar o rechazar un nombramiento en específico.

Al dar clic en “Verifica” o “Rechaza” se invoca al archivo verificaUapa.java que realizará una actualización de la tabla oficio_asignacion (como se puede observar en la figura 4.16

en la variable `qrylf1` se guarda la sentencia que actualizará la información del nombramiento en la base de datos y mediante la función `query.actualiza(qrylf1)` se ejecuta la actualización)para poner el status que corresponda al nombramiento de acuerdo a si es rechazado (status 1) o se aprueba la propuesta (status 2) y regresa el control a la Vista `revison-uapa.jsp`, a continuación se muestra en la figura 4.16 el código que permite realizar lo anterior:

```

querys query = new querys();
String qrylf1 = "select comentarios from oficio_asignacion where profesor_idprofesor="+prof+
    " and cicloescolar_idcicloescolar="+ciclo+" and profesor_idnombramiento="+idnom;
try{
    ResultSet rsqry = query.consultar(qrylf1);
    rsqry.next();
    comentario+="\n" + rsqry.getString("comentarios");

    qrylf1 = "update oficio_asignacion set comentarios='\"+comentario+"\', fecharevision='\"+
        now+"\', status='1' where profesor_idprofesor="+prof+" and cicloescolar_idcicloescolar="+ciclo+
        " and profesor_idnombramiento="+idnom;
    query.actualiza(qrylf1);
} catch (Exception e) {System.out.println("Error en update verificaUapa "+e);}
query.closeq();
response.sendRedirect("errorValida.jsp?type="+type+"&idr="+idr+"&s="+idsecc+"&d="+iddepto+"&sta="+sta+"&p="+prof);
} else { //aquí es donde aprueba - status = 2
    Date now = Calendar.getInstance().getTime();
    querys query = new querys();
    comentario = "Nombramiento REVISADO - OK";
    String qrylf1 = "update oficio_asignacion set comentarios='\"+comentario+"\', fechaok='\"+
        now+"\', status='2' where profesor_idprofesor="+prof+" and cicloescolar_idcicloescolar="+ciclo+" and
        profesor_idnombramiento="+idnom;
    try{query.actualiza(qrylf1);
    } catch (Exception e) {System.out.println("Error en update verificaUapa "+e);}
    query.closeq();
    response.sendRedirect("errorValida.jsp?type=3&idr="+idr+"&s="+idsecc+"&d="+iddepto+"&sta="+sta+"&p="+prof);
}
} catch (Exception e){System.out.println("error verificaUapa general");}
}

```

Figura 4.16 verificaUapa.java, para aprobar o rechazar un nombramiento en específico y actualizar su status.

Por último el usuario UAPA podrá enviar la propuesta de nombramientos académicos correctos al Consejo Técnico, esto es posible dando clic en la liga Enviar dentro del archivo `revison-uapa.jsp` como se muestra en la figura 4.17, la cual realiza un procedimiento similar a la aprobación y rechazo del nombramiento, en el que solo se actualiza la tabla `oficio_asignacion` cambiando el status a 4, el cual significa que esta propuesto a Consejo Técnico para su aprobación.

	Profesor	Nombramiento	Hrs	Mov	Status Revisión	Env. CT
1	DIAZ CELAYA SAYDA	Prof Asig A, i	6.0	OTRO NOMBRAMIENTO	Revisado - OK	Enviar
	DDCV/HMM/001/12/2011	Del 2012-01-30	Al 2012-08-05	Diseño y Comunicación Visual	AUDIOVISUAL FOTOGRAFIA Y MULTIMEDIA	
2	GARCÍA GONZÁLEZ MAURICIO RUBEN	Prof Asig A, i	4.0	OTRO NOMBRAMIENTO	Revisado - OK	Enviar
	DDCV/HMM/291/02/2011	Del 2012-01-30	Al 2012-08-05	Diseño y Comunicación Visual	AUDIOVISUAL FOTOGRAFIA Y MULTIMEDIA	
3	JACINTO ROBLES ELOY	Prof Asig A, i	8.0	OTRO NOMBRAMIENTO	Revisado - OK	Enviar
	DDCV/HMM/032/10/2011	Del 2012-01-30	Al 2012-08-05	Diseño y Comunicación Visual	DISEÑO DE ILUSTRACIÓN + SIMBOLOGÍA Y DIS SOP. TRIDI	
4	MORALES MEJIA HÉCTOR RAÚL	Prof Asig A, i	22.0	PRÓRROGA	Revisado - OK	Enviar
	DDCV/HMM/020/10/2011	Del 2012-01-30	Al 2012-08-05	Diseño y Comunicación Visual	DISEÑO DE ILUSTRACIÓN + SIMBOLOGÍA Y DIS SOP. TRIDI	
5	PIÑA MORALES VERONICA	Prof Asig A, i	6.0	OTRO NOMBRAMIENTO	Revisado - OK	Enviar
	DDCV/HMM/003/01/2012	Del 2012-01-30	Al 2012-08-05	Diseño y Comunicación Visual	DISEÑO DE ILUSTRACIÓN + SIMBOLOGÍA Y DIS SOP. TRIDI	
6	PIÑA PEREZ ESTEBAN	Prof Asig A, i	4.0	OTRO NOMBRAMIENTO	Revisado - OK	Enviar
	DDCV/HMM/002/12/2011	Del 2012-01-30	Al 2012-08-05	Diseño y Comunicación Visual	AUDIOVISUAL FOTOGRAFIA Y MULTIMEDIA	
	SANDOVAL VALLE MARCO ANTONIO	Prof Carr Asoc C tc, i	40.0	PRÓRROGA	Revisado - OK	Enviar

Figura 4.17 revision-uapa.jsp, para proponer un nombramiento específico al Consejo Técnico y actualizar su status.

4.5.Pruebas

Como se menciona en el capítulo 2 las pruebas son parte fundamental en el ciclo de desarrollo de software, ya que durante esta actividad se realizan una serie de pruebas que permiten validar que el sistema realmente cumple las necesidades del cliente, para este sistema se realizaron pruebas tanto por el equipo de desarrollo como por los usuarios finales:

Pruebas de unitarias: Estas pruebas se realizaron con el objetivo de aislar cada parte del programa y mostrar que las partes individuales son correctas (por ejemplo una función para saber si las actividades de docencia y de apoyo asignadas a un profesor no exceden las 8 horas diarias).

Pruebas de integración: Se realizaron una serie de pruebas que permitieron determinar si la integración del módulo de UAPA se acopla a los ya existentes en el sistema, ya que trabaja directamente con el módulo de los departamentos académicos y con el módulo de Consejo Técnico, por lo tanto se verifico que la interacción y la información de entrada y salida módulos fuera correcta.

Pruebas alfa y beta: Se realizaron pruebas alfa y beta para que por medio de ellas se pudieran descubrir errores que sólo pudieran ser detectados por el usuario final

- **Prueba alfa:** Se lleva a cabo, por un cliente, en el lugar de desarrollo. Se usa el software de forma natural con el desarrollador como observador del usuario y registrando los errores y problemas de uso. Las pruebas alfa se llevan a cabo en un entorno controlado.

- **Prueba beta:** Se llevan a cabo por los usuarios finales del software en los lugares de trabajo de los clientes. A diferencia de la prueba alfa, el desarrollador no está presente normalmente. Así, la prueba beta es una aplicación en vivo del software en un entorno que no puede ser controlado por el desarrollador. El cliente registra todos los problemas que encuentra durante la prueba beta e informa al desarrollador.

El desarrollo de estas pruebas fue con el objetivo de detectar operaciones no previstas o errores, para darles solución lo antes posible y hacer que el sistema cumpla al 100% los requerimientos de los usuarios.

4.6.Capacitación

Como en cualquier producto de software, es necesario conocer su funcionamiento para obtener el mejor aprovechamiento del mismo, por ello se realizaron varias sesiones de capacitación a los usuarios del sistema, estas sesiones se dividieron de la siguiente manera:

Capacitación general: En estas sesiones se mostro el funcionamiento general del sistema para que todos los actores del sistema conozcan los procesos que se llevan a cabo dentro del mismo y en cuales son participes (usuarios: Coordinadores, Departamentos y Secciones Académicas, UAPA, Consejo Técnico, Departamento de Personal, etc.).

Capacitación por procesos: En estas sesiones se capacitó a los actores que en sus procesos internos afectan los procesos de los demás usuarios, tal es el caso de los usuarios del módulo de UAPA, los cuales se capacitaron junto con los usuarios de los módulos de Departamentos y Secciones Académicas, Consejo Técnico y el Departamento de personal, ya que son los actores principales para llevar a cabo la contratación de un académico.

Finalmente se realizó la capacitación exclusiva de los usuarios del módulo de UAPA para familiarizarlos con el sistema y validando que el sistema cumpliera al 100% con sus requerimientos.

4.7.Liberación

Una vez que se concluyeron las pruebas al sistema y la capacitación, se entregó de manera formal el nombre de usuario y contraseña a los diferentes usuarios, este proceso se realizó permitiendo al usuario capturar su propio nombre de usuario y contraseña, para que quedaran cifrados los datos y sólo él conociera esa información.

Actualmente el sistema se encuentra en funcionamiento para capturar la información de los semestres 2011-1 y 2011-2.

Conclusiones

El desarrollo del Sistema de Asignaciones Docentes FESC ha contribuido de manera muy importante a mi desarrollo profesional, al permitirme identificar y resaltar los puntos que se tienen que considerar para llevar a cabo la implementación exitosa de un sistema de información.

Dentro de las experiencias que considero con mayor importancia dentro del desarrollo de proyectos se encuentra el realizar un análisis minucioso a todos los procesos que se lleven a cabo, con el fin de detectar de manera precisa las necesidades reales de los usuarios, ya que el no realizar de manera correcta la toma de requerimientos generará un obstáculo en el desarrollo de sus actividades, dando como resultado un gasto en lugar de una inversión. Otro aspecto importante y necesario es involucrar a los usuarios en el proceso de implementación de los sistemas, de tal manera que sepan que es lo que esperan del sistema y que es lo que no esperan de él, ello ayudará a obtener un mejor nivel de aceptación del mismo y por lo tanto las aportaciones de los usuarios serán mejores, logrando un beneficio mayor tanto para la empresa o institución como de los propios usuarios.

Otro punto que considero importante es llevar a cabo una buena capacitación a los usuarios de los sistemas, ya que si desarrollamos e implementamos correctamente un sistema pero no le damos herramientas a los usuarios para trabajar en el, es muy probable que todo el trabajo realizado sea en vano o simplemente no generara los resultados esperados.

El objetivo primordial que se desea lograr con el sistema es el brindar un servicio que eficiente sus procesos disminuyendo el tiempo de realización de los mismos, lo cual se cumple al 100% ya que se detecto una disminución de errores y tareas rutinarias en todas las áreas, permitiendo dedicar más tiempo a tareas productivas más que a las administrativas, cumpliendo así con las necesidades de los usuarios. Todo ello no involucra que no se tengan oportunidades de mejora, las cuales se que podrán cubrir mediante la actualización constante del sistema para que siga manteniendo si vigencia y funcionalidad.

Apéndice A

Nombre de la tabla	Atributos	Tipo	Descripción
Actividadapoyo	idactividadapoyo	int	Id de la actividad de apoyo
	rango	varchar(100)	PACIVE, PAPITT, Apoyo Académico, etc.
	tipo	char(1)	A, B, C
	actividadapoyo	varchar(500)	Nombre de la actividad de apoyo
	hmin	float	Mínimo de horas que se pueden asignar a esa actividad
	hmax	float	Máximo de horas que se pueden asignar a esa actividad
	abreviado	varchar(3)	Abreviatura del rango
actividadapoyo_profesor	idactividadapoyo	int	Id de la actividad de apoyo
	profesor_idnombramiento	int	Id del nombramiento del profesor
	profesor_idprofesor	int	Id del profesor
	cicloescolar_idcicloescola	int	Id del ciclo escolar
	usuario_idusuario	int	Id del usuario
	comentarios	text	Descripción de la actividad que realiza
	ubicacion	varchar(200)	Ubicación donde se realiza la actividad de apoyo
	lu_ini	int	Horario
	lu_fin	int	Horario
	ma_ini	int	Horario
	ma_fin	int	Horario
	mi_ini	int	Horario
	mi_fin	int	Horario
	ju_ini	int	Horario
	ju_fin	int	Horario
	vi_ini	int	Horario
	vi_fin	int	Horario
	sa_ini	int	Horario
	sa_fin	int	Horario
htotales	float	Total de horas asignadas a la actividad de apoyo	

			indica si la actividad de apoyo corresponde al semestre o al intersemestre (1,0)
	Intersemestral	int	
Apertura	idapertura	int	Id único de la apertura
	idcicloescolar	int	Id único del ciclo escolar
	idmodulo	int	Módulo al que se dará apertura de manera general (secciones, coordinadores, deptos, etc.).
	fechaapertura	datetime	Fecha de apertura para el módulo
	fechacierre	datetime	Fecha de cierre para el módulo
	usuario	char(18)	Identificador del usuario a quien se realiza la apertura
	Carrera	idcarrera	int
carrera		varchar(200)	Nombre de la carrera
Catedra	idcatedra	int	Identificador único de la cátedra
	tipo	varchar(100)	Tipo de la cátedra (PACIVE, PAPITT, etc.)
	identificador	varchar(100)	Identificador de la cátedra
	catedra	varchar(500)	Nombre de la cátedra
	area	varchar(500)	Área en la que se desarrolla la cátedra
	responsable	varchar(500)	Responsable de la cátedra
	finicio	datetime	Fecha de inicio de la cátedra
	ffin	datetime	Fecha de termino de la cátedra
catedra_participante	idcatedra	int	Identificador único de la cátedra
	idprofesor	int	Identificador único del profesor
Causa	idcausa	int	Id único de la causa
	idmovimiento	int	Id del movimiento
	nombre_causa	varchar()	Nombre de la causa que tendrá el nombramiento de algún profesor
	idcicloescolar	int	Id único del ciclo escolar

Cicloescolar	finicio	timestamp	Fecha en que inicia el ciclo escolar
	ftermino	timestamp	Fecha en que termina el ciclo escolar
	cicloescolar	char(4)	Año del ciclo escolar (2011)
	semestre	char(1)	Semestre del ciclo escolar (1,2)
Departamento	iddepartamento	int	Identificador único del departamento
	departamento	char(100)	Nombre del departamento
	jefedepartamento	char(150)	Responsable del departamento
Edificio	idedificio	Int	Id único del edificio
	campo	char(18)	Campo en el que se encuentra ubicado el edificio
	nom_edificio	char(40)	Nombre del edificio
grupo_practico_transicion	idgrupo_transicion	Int	Id del grupo del cual es su parte practica y que se encuentra en grupo_transicion
	idgrupo_practico_transicion	char(18)	Identificador del grupo practico, es consecutivo según el número de grupos practico se abran para un grupo teórico.
	consecutivo	char(18)	Letra que se asigna al grupo practico, (A, B, C, etc.)
	lu_ini	char(18)	Horario
	lu_fin	char(18)	Horario
	ma_ini	char(18)	Horario
	ma_fin	char(18)	Horario
	mi_ini	char(18)	Horario
	mi_fin	char(18)	Horario
	ju_ini	char(18)	Horario
	ju:fin	char(18)	Horario
	vi_ini	char(18)	Horario
	vi_fin	char(18)	Horario
	sa_ini	char(18)	Horario
	sa_fin	char(18)	Horario
	idsalon	Int	Id del salón
	idedificio	Int	Id del edificio
	repetido	char(18)	

	idcicloescolar	Int	Id del ciclo escolar	
grupo_profesor	profesor_idprofesor	Int	Id del profesor	
	idgrupo_transicion	Int	Id del grupo	
	grupo_idgrupo_practico	Int	El id del grupo práctico en caso de que el grupo sea práctico y 0 si es teórico	
	profesor_idnombramiento	Int	Id del nombramiento	
	idcicloescolar	Int	Id del ciclo escolar	
	htotales	float	Horas asignadas al profesor en ese grupo	
	compartido	char(1)	Indica si el grupo ha sido partido para ser asignado en dos nombramientos del profesor o en nombramientos de distintos profesores	
	titular	char(1)	Indica quien es el titular del grupo	
	usuario_idusuario	Int	Usuario que realizo el movimiento	
	materia_idmateria	Int	Id de la materia	
	carrera_idcarrera	Int	Id de la carreras	
grupo_profesor_compartido	profesor_idprofesor	Int	Id del profesor	
	idgrupo_transicion	Int	Id del grupo	
	grupo_idgrupo_practico	Int	Id del grupo practico	
	profesor_idnombramiento	Int	Id del nombramiento	
	idcicloescolar	Int	Id del ciclo escolar	
	usuario_idusuario	Int	Usuario que realiza el movimiento	
				Horario del grupo, este horario es nuevo debido a que el grupo a sido partido y el horario es distinto, por lo tanto en la tabla grupo profesor los que tengan un grupo con un 1 en el campo compartido, deberán acceder al horario desde esta tabla.
	lu_ini	Int		
	lu_fin	int		Horario
	ma_ini	int		Horario
	ma_fin	int		Horario
	mi_ini	int		Horario
	mi_fin	int		Horario

	ju_ini	int	Horario
	ju_fin	int	Horario
	vi_ini	int	Horario
	vi_fin	int	Horario
	sa_ini	int	Horario
	sa_fin	int	Horario
	htotales	float	Horas totales asignadas en este grupo
grupo_transicion	idgrupo_transicion	int	Id del grupo
	grupo_transicion	varchar(5)	Nombre del grupo
	idcicloescolar	int	Id del ciclo escolar
	idmateria	int	Id de la materia
	saturacion	int	Saturación del grupo
	idcarrera	int	Id de la carrera
	turno	int	Turno (matutino, vespertino)
	tipo	int	1 Teórico 2 Teórico-Práctico 3 Práctico
	lu_ini	int	Horario
	lu_fin	int	Horario
	ma_ini	int	Horario
	ma_fin	int	Horario
	mi_ini	int	Horario
	mi_fin	int	Horario
	ju_ini	int	Horario
	ju_fin	int	Horario
	vi_ini	int	Horario
	vi_fin	int	Horario
	sa_ini	int	Horario
	sa_fin	int	Horario
	idsalon	int	Salón
	idedificio	int	Edificio
	equivalente	char(1)	
	semestre	int	Semestre en que se imparte la materia
	aprobado	char(1)	Estado que guarda el grupo
	repetido	char(1)	
		idmateria	int
idcarrera		int	Identificador único de la carrera
idseccion		int	Id de la sección a la que pertenece la carrera

Materia	iddepartamento	int	Id del departamento al que pertenece la carrera
	materia	varchar(100)	Nombre de la materia
	semestre	int	Semestre en el que se imparte la materia
	creditos	int	Créditos de la materia
	horasteoricas	float	Horas teóricas de la materia según plan de estudios
	horaspracticas	float	Horas practicas de la materia según plan de estudios
	optativa	char(1)	Indica si la materia es optativa (1,0)
materia_compartida	idmateriaa	int	Identificador de la materia origen
	idmateriab	int	Identificador de la carrera origen
	idmateria	int	Identificador de la materia compartida
	idcarrera	int	Identificador de la carrera a la que pertenece la materia compartida
Modulo	idmodulo	int	Identificador del módulo
	modulo	varchar(45)	Nombre del módulo
Movimiento	idmovimiento	int	Id del movimiento
	nombre_movimiento	varchar()	Nombre del tipo de movimiento (alta, baja, etc.)
	idnombramiento	int	Id único del nombramiento
	nombramiento	varchar(45)	Nombre del nombramiento
	maxgrupos	int	Máximo de grupos que puede tener un profesor según el nombramiento
	minhoras	int	Mínimo de horas de docencia según el nombramiento
	maxhoras	int	Máximo de horas de docencia según el nombramiento
	clavenombramiento	varchar()	Clave tipo texto del nombramiento
	nombramientocatalogo	varchar()	Nombre del nombramiento
	nivel	varchar()	Nivel del nombramiento (A,B,C, etc.)

nombramiento	contrato	varchar()	Tipo de contrato (Interino o definitivo)
	tiempo	varchar()	Cantidad de horas asignadas al nombramiento (horas, mt o tc), por horas es null en esta tabla por que el usuario las introduce cuando hace la relación profesor-nombramiento en la tabla de prof_nombramiento
	partida	int	Partida para el tipo de nombramiento
	nombramientomayor	int	Nombramiento mayor
	nombre	varchar()	Nombre corto del nombramiento
	ltn	varchar()	
	oficio_Asignacion	profesor_idprofesor	int
cicloescolar_idcicloescolar		int	Id del ciclo escolar de la asignación que se enviara a UAPA
profesor_idnombramiento		int	Id único del profesor
oficio		varchar()	Numero de oficio
usuario_idusuario		int	Id del usuario que realizo el movimiento
Profesor	idprofesor	int	Identificador único del profesor
	fecha_ingreso_unam	datetime	Fecha de ingreso a la UNAM
	num_empleado	varchar()	Numero de trabajador asignado por la UNAM
profesor_depto_seccion	idprofesor	int	Identificador único del profesor
	idseccion	int	Identificador único de la sección
	iddepartamento	int	Identificador único del departamento al que pertenece la sección
	fecha_ingreso_seccion	timestamp	Fecha en que ingreso a la sección
profesor_direccion	idprofesor	int	Identificador único del profesor
	estado	varchar(20)	Estado de la Republica donde vive
	calleynumero	varchar(100)	Calle y numero
	colonia	varchar(70)	Colonia
	email	varchar(70)	Correo electrónico

	teldomicilio	varchar()	Teléfono de domicilio
	teloficina	varchar()	Teléfono de oficina
	extension	varchar()	Extensión
	cp	varchar()	Código postal
	municipio	varchar()	Delegación o Municipio
	telcelular	varchar()	Teléfono celular
profesor_generales	idprofesor	int	Identificador único del profesor
	nombre	char(45)	Nombre del profesor
	apaterno	char(45)	Apellido paterno
	amaterno	char(45)	Apellido materno
	rfc	char(45)	RFC del profesor
	homoclave	char(3)	Homoclave 3 dígitos
	curp	char(45)	CURP del profesor
	nacionalidad	varchar()	Nacionalidad del profesor
	estado_civil	varchar()	
	genero	varchar()	Genero del profesor
	fecha_nacimiento	timestamp	Fecha de nacimiento
profesor_grado_acad	idprofesor	int	Identificador único del profesor
	grado_academico	varchar()	Grado académico al momento del registro
	area_especialidad	varchar()	Área de especialidad
	fecha_grado	Timestamp	Fecha de obtención de grado
	sni	varchar()	Nivel del SNI en caso de tener
	fecha_ingreso_sni	timestamp	Fecha de ingreso al SNI
	promedio	float	Promedio en caso de no ser titulado
	creditos	float	Numero de créditos en caso de no ser titulado
profesor_nombramiento	profesor_idnombramiento	Int	
	idprofesor	Int	Identificador único del profesor
	fecha_movimiento	timestamp	Fecha en que se realizado el movimiento
	oficio_movimiento	varchar()	Oficio por el cual se crea el nombramiento
	numero_horas	float	Número de horas asignadas al nombramiento
	movimiento_aprobado	varchar()	Identificador de el estado del movimiento (aprobado o no

			aprobado)
	fecha_mov_aprob	timestamp	Fecha de aprobación
	oficio_mov_aprob	varchar()	Oficio del movimiento de aprobación
	idcausa	int	Identificador de la causa
	fechainicio_nomb	timestamp	Fecha de inicio para el nombramiento
	fechatermino_nomb	timestamp	Fecha de termino para el nombramiento
	idusuario	int	Usuario que realizo el movimiento
	fecha_reg_nomb	timestamp	
	vigente	varchar()	Vigencia del nombramiento
	estadoct	varchar()	Estado de CT
	sesion	varchar()	Sesión de CT
	reunion	varchar()	Reunión de CT
	iddepartamento	int	Identificador de departamento
	idseccion	int	Id de la sección
	idcicloescolar	int	Id del ciclo escolar
	idnombramiento	int	Id del nombramiento
	idmovimiento	int	Id del movimiento
salon	idsalon	int	Id del salón
	idedificio	int	Id del edificio
	nombre_salon	char(40)	Nombre del salón
	capacidad	int	Capacidad de alumnos que permite el salón
seccion	idseccion	int	Identificador único de la sección
	iddepartamento	int	Identificador único del departamento al que pertenece la sección
	seccion	varchar(100)	Nombre de la sección
Usuario	idusuario	int	Identificado único del usuario
	usuario	varchar(50)	Nombre con que el usuario ingresara al sistema (SHA1)
	clave	varchar(50)	Contraseña de acceso al sistema (SHA1)
	nombra	varchar(50)	Nombre del usuario
	apaterno	varchar(50)	Apellido paterno
	amaterno	varchar(50)	Apellido materno
	extension	int	Numero de extensión dentro de la UNAM

	email	varchar(50)	Correo electrónico
	idmodulo	int	Módulo al que pertenece el usuario según su contratación

Bibliografía

1 Misión de la Facultad de Estudios Superiores Cuautitlán. Disponible en la WWW (<http://www.cuautitlan.unam.mx/misionyvision.html>)

2 Visión de la Facultad de Estudios Superiores Cuautitlán. Disponible en la WWW (<http://www.cuautitlan.unam.mx/misionyvision.html>)

3 Organización institucional de la Facultad de Estudios Superiores Cuautitlán. Disponible en la WWW (<http://www.cuautitlan.unam.mx/institucional/>)

4 Whitten Bentley (2008). Análisis de sistemas, diseño y métodos, 7^a edición, Ed McGraw-Hill. México.

5 James A. Senn (1992). Análisis y diseño de sistemas de información, 2^a edición, Ed McGraw-Hill. México.

6 Definición de ciclo de vida de software ISO/IEC 12207. Disponible en la WWW (http://es.wikipedia.org/wiki/ISO/IEC_12207)

7 Ian Sommerville. Ingeniería de Software, 7^a edición, Ed Prentice Hall. México (2005).

8 Modelos de proceso o ciclo de vida. Disponible en la WWW (http://es.wikipedia.org/wiki/Software#Modelos_de_proceso_o_ciclo_de_vida)

9 Sistemas distribuidos. Disponible en la WWW (<http://www.monografias.com/trabajos16/sistemas-distribuidos/sistemas-distribuidos.shtml>)

10 Sistemas distribuidos. Disponible en la WWW (http://es.wikipedia.org/wiki/Computaci%C3%B3n_distribuida)

11 Sistemas distribuidos. Disponible en la WWW (<http://www.monografias.com/trabajos16/sistemas-distribuidos/sistemas-distribuidos.shtml#APLICAC>)

12 Sistemas distribuidos. Disponible en la WWW (<http://www.augcyl.org/?q=glol-intro-sistemas-distribuidos>)

13 Arquitectura Cliente-Servidor. Disponible en la WWW (<http://es.wikipedia.org/wiki/Cliente-servidor>)

14 Arquitectura Cliente-Servidor. Disponible en la WWW (http://es.wikipedia.org/wiki/Cliente_%28inform%C3%A1tica%29)

15 Arquitectura Cliente-Servidor tres capas. Disponible en la WWW

(<http://tallerbd.wikispaces.com/ARQUITECTURA+CLIENTE-SERVIDOR+DE+3+CAPAS>)

16 Arquitectura Cliente-Servidor tres capas. Disponible en la WWW
(<http://es.scribd.com/doc/53015645/5/Arquitecturas-Cliente-Servidor-de-tres-capas>)

17 Arquitectura Cliente-Servidor tres capas. Disponible en la WWW
(<http://oposicionestic.blogspot.com/2011/06/arquitectura-cliente-servidor.html>)

18 Arquitectura Cliente-Servidor multi-capas. Disponible en la WWW
(http://en.wikipedia.org/wiki/Multitier_architecture)

19 Base de datos. Disponible en la WWW
(<http://www.monografias.com/trabajos7/bada/bada.shtml>)

20 Base de datos. Disponible en la WWW
(http://upload.wikimedia.org/wikipedia/commons/6/6f/Base_de_datos.pdf)

21 Arquitectura de base de datos. Disponible en la WWW
(<http://es.scribd.com/doc/5559477/Fundamentos-de-Base-de-datos>)

22 David M. Kroenke (2003). Procesamiento de bases de datos – Fundamentos, diseño e implementación, 8ª edición, Ed. Prentice Hall. México.

23 Raghu Ramakrishnan, Johannes Gehrke (2000). Database Management Systems, 2ª edición, Ed MacGraw-Hill. México.

24 David M. Kroenke, David J. Auer (2010). Database Concepts, 4ª edición, Ed Prentice Hall. México.

25 Catherine M. Ricardo (2010). Bases de Datos, 1ª edición, Ed MacGraw-Hill. México.

26 Sistema operativo. Disponible en la WWW
(http://es.wiktionary.org/wiki/sistema_operativo)

27 Servidor de aplicaciones. Disponible en la WWW
(http://es.wikipedia.org/wiki/Servidor_de_aplicaciones)

28 Java. Disponible en la WWW
(http://es.wikipedia.org/wiki/Java_%28lenguaje_de_programaci%C3%B3n%29)

29 Java. Disponible en la WWW
(<http://docs.oracle.com/cd/E19830-01/819-5893/galmu/index.html>)