UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SISTEMA DE ASIGNACIÓN DE CALIFICACIONES Y REVALIDACIONES DE LABORATORIOS PARA EL DEPARTAMENTO DE INGENIERÍA EN COMPUTACIÓN

T E S I S

QUE PARA OBTENER EL TÍTULO DE:
INGENIERO EN COMPUTACIÓN

P R E S E N T A:

LÓPEZ GARCÍA JORGE LUIS

DIRECTOR DE TESIS
ING. FILIBERTO MANZO GONZÁLEZ

MÉXICO, D. F. NOVIEMBRE, 2011

Agradecimientos

En primer lugar debo ofrecer una disculpa por no estar ahí como debí dejando de ser un buen hijo, hermano y amigo por ser egoísta en la búsqueda de concluir esta tesis para finalmente subir un peldaño tan importante y trascendental en mi vida.

Gracias a todos los que contribuyeron a esta causa brindándome ánimo, apoyo y una sonrisa en cada momento de oscuridad.

A ese ser que me ha cuidado y a las personas que quiero, por darme vida para llegar hasta aquí.

Gracias.

A la Universidad Nacional Autónoma de México a la que adoro tanto, la Facultad de Ingeniería y sus profesores que me formaron con conocimientos, razonamiento y valores para contribuir al beneficio de la sociedad.

A mi director de tesis por su confianza, consejos, asesoría, tiempo y amistad.

A mis sinodales, personas de gran conocimiento y calidad humana a quienes aprecio y respeto por todas las enseñanzas que han compartido conmigo.

A mi madre por ser mi fortaleza, consejera, apoyo, motivación y una fuente de admiración y respeto durante toda mi vida. Gracias por ser un ejemplo de esfuerzo y rectitud, por darme la oportunidad de tener una educación que me haga crecer profesionalmente y formarme con buenas costumbres y sentimientos.

A mi padre, aunque ya no está con nosotros su espíritu sigue conmigo alentándome a ser mejor. Estaré eternamente agradecido por su constante apoyo en mi paso académico siendo partícipe de cada pequeño pero valioso logro con sus conocimientos e inspiración. Gracias por guiarme a ser la persona que soy ahora fomentando en mi la dedicación, el esfuerzo y la responsabilidad.

A mi hermana Lety por soportar mi mal genio y aún así apoyarme con una sonrisa o un detalle. Por desvelarse conmigo o esperar para darme las buenas noches. Por acompañarme desde pequeño y ser una segunda madre.

Agradecimientos

A mis hermanos Jorge Mario y José Vidal por su interés en mi salud, crecimiento, consentirme y estar ahí cuando los he necesitado.

A mi hermana María Luisa que pese a su discapacidad me ha brindado su cariño, abrazos y ternura.

A mi tía Leonor por ser pieza crucial de mi educación y crecimiento personal con su respaldo en cada momento difícil y otros de felicidad.

A mi amigo Carlos por ser siempre un gran apoyo, gran amigo y persona, por estar ahí en el momento más difícil y ser como mi hermano mayor.

A mis grandes amigos que me acompañaron en esta divertida aventura en la Facultad de Ingeniería, **Eric, Raúl, Christian y Danice** con quienes compartí emociones y tristezas, logros y fracasos.

A Karlita por su amistad, cariño y apoyo en los días de desesperación y a la vez de alegría, por su paciencia y comprensión.

A Alfredo por su incondicional asesoría, enseñanzas y valiosos conocimientos sin los cuales el sistema no hubiera sido posible.

A mis amigas de AFG **Flor y Hayde** quienes han estado pendientes de mi avance y conclusión de esta tesis con mensajes de apoyo.

A mis amigas del Departamento, comenzando por **Mony** por permitirme ser su amigo y conocer a su maravillosa familia, así mismo por su interés y motivación en este largo camino.

A Mayelly por su infinita compresión, apoyo y amistad. Su confianza en mí fue definitiva para salir de la sombra con cada frase de motivación. Gracias por ser el factor determinante para llegar a esta meta.

A todos ustedes les estaré siempre agradecido y también a aquellas personas que en su momento fueron importantes y se quedaron en el camino.

"Piensa, cree, sueña y atrévete."

Walt Disney

Prólogo

El concepto de computadora como herramienta de trabajo completa e independiente ha perdido peso en los últimos años, trasladando su verdadero valor a la capacidad de conectarse a una red de comunicaciones donde interactúe con otros equipos electrónicos y redes informáticas.

Las aplicaciones tienden hacia la Web automatizando actividades en las cuales ya no es requerida la presencia del usuario pero si una conexión a Internet. Se busca que el usuario evite pérdidas de tiempo en traslados, filas y en su lugar envíe la información a través de la red en la comodidad de su hogar u oficina.

Por lo anterior, así como por el número creciente de personas con una conexión a Internet y la diversidad de portales para empresas, productos y departamentos gubernamentales, el número de sistemas para registro de datos y trámites también ha crecido.

El sistema de asignación de calificaciones y revalidaciones de laboratorios para el Departamento de Ingeniería en Computación expuesto en esta tesis se adhiere al enfoque Web antes mencionado para permitir a los usuarios a través de la red cumplir con objetivos como la asignación de calificaciones de laboratorios y la obtención de las mismas considerando también el beneficio de la revalidación.

Así mismo permite la gestión de la información almacenada en una base de datos de forma remota para que finalmente los profesores de teoría puedan evaluar a tiempo.

La tesis se compone de cuatro capítulos a fin de cubrir las etapas del desarrollo del sistema, partiendo del capítulo 1 donde se tratan la problemática a solucionar, las organizaciones involucradas y fundamentos teóricos de ingeniería de software, programación, bases de datos y sistemas operativos. El capítulo 2 trata del análisis del sistema a desarrollar tomando en cuenta los requisitos, riesgos y las herramientas de software a emplear. En el capítulo 3 se muestra el diseño del sistema a través de diagramas con la representación de sus funciones y el manejo de la información. Finalmente, el capitulo 4 cubre la implementación del sistema con ayuda de tecnologías para la presentación al usuario, la programación de los módulos y la persistencia de los datos.

Índice

Capítulo 1

Introducción

1.1 Definición del problema

El Departamento de Ingeniería en Computación desempeña un papel importante como enlace entre los estudiantes, profesores, universidades y el campo laboral, coordinando aspectos de la carrera tales como plan de estudios, intercambios universitarios, investigación, programación de horarios, así como la contratación de personal docente capaz y comprometido con el aprendizaje y superación de los estudiantes.

Se encarga también de trámites propios de dichas actividades como la recepción de documentos de profesores de nuevo ingreso y los procedimientos para la generación de nombramientos de todos los profesores, ayudantes de profesor y técnicos académicos del Departamento a partir de una base de datos.

La cantidad de información manejada por el personal aumenta en complejidad y responsabilidad al administrar el proceso de recepción de calificaciones de los laboratorios de 8 asignaturas que deben ser organizadas y enviadas a cada profesor de teoría, que van de los 6 a los 34 grupos cada una con un aproximado de 16 a 55 alumnos por grupo. Además deben ser incorporadas las calificaciones de semestres lectivos previos como parte del proceso de revalidación que tiene como fin brindar un apoyo al estudiante que acreditó el laboratorio pero no así la teoría.

De esta forma el alumno regular y el que está en artículo 33 del Reglamento General de Inscripciones (Alumno Sin Derecho a ReInscripción o ASDRI) pueden cursar de nueva cuenta la teoría conservando su calificación aprobatoria de laboratorio si es el caso con lo que se reduce la demanda de grupos de laboratorio, asimismo será válida para los alumnos que presenten examen extraordinario quienes también deben tener el laboratorio aprobado.

1.1.1 Panorama actual

Desde hace algunos años el personal del Departamento ha desarrollado y adaptado un sistema que tiene como fin facilitar algunas etapas del proceso de recepción, revalidación y envío de calificaciones de laboratorio, sin embargo no es del todo fiable y es propenso a errores humanos.

En resumen, el desarrollo actual se da de la siguiente forma:

En la última semana de clases de cada semestre el profesor de laboratorio debe enviar las evaluaciones de sus estudiantes con información específica: nombre, número de cuenta, carrera, grupo de teoría y calificación obtenida. Parte de esta información les es entregada al descargar su lista de alumnos inscritos del portal de la Unidad de Servicios de Cómputo Administrativo (USECAD) que brinda este servicio a los profesores de la Facultad de Ingeniería.

El primer obstáculo se presenta cuando estos profesores no entregan en tiempo y forma, al no cumplir los requisitos del formato ni las fechas establecidas, lo que lleva a revisar minuciosamente documento por documento y contactar constantemente al docente para solicitar su evaluación, siendo ésta una actividad muy tediosa y tardada.

Si esto sucede los profesores de teoría se ven afectados en su cronograma al retrasar su evaluación final en la que contemplan cierto porcentaje para el laboratorio y para la cual deben tomar en cuenta además que es requisito indispensable aprobar el laboratorio para acreditar la asignatura.

Una vez recibidas las listas de todos los grupos de una asignatura y verificado su formato se pasan por una macro programada en Microsoft Excel, la cual mediante comandos de Visual Basic y sentencias de SQL agrupa las calificaciones y las exporta a un archivo por asignatura con sus correspondientes grupos.

Como parte imprescindible de la metodología actual está el descargar en las primeras semanas de iniciado el semestre las listas de alumnos inscritos incluidos ASDRI del portal de USECAD para cada grupo de teoría. A dichos archivos se les da un tratamiento especial a fin de adaptarlo al formato requerido por la macro, que una vez ejecutada y generado el archivo se extrae la lista con las calificaciones finales y se envía mediante correo electrónico al profesor de teoría correspondiente.

Sin embargo, pueden presentarse casos donde el número de cuenta, nombre o calificación no estén escritos correctamente y al no contar con un módulo de verificación no se produce ningún aviso de estos errores y pasan desapercibidos.

Otra carencia del método actual consiste en la opción de búsqueda de alumno de forma individual dado que en ocasiones deben realizarse correcciones de calificación, afrontando esta situación con una revisión de cada uno de los archivos de forma manual lo que representa una pérdida de tiempo y esfuerzo para el personal.

También destaca la falta de seguridad y restricciones para el acceso a la información, donde cualquiera con cercanía al equipo donde ésta se almacena puede realizar alteraciones a las calificaciones sin generarse ningún reporte de tal evento.

Lo anterior constituye un largo y complejo proceso de muchos pasos, que recibe información de terceros y se convierte en factor de error al no ser entregada como se requiere, propenso a fallas por parte del personal del Departamento de Computación y vulnerable a alteraciones, lo cual repercutiría directamente en la evaluación y avance de los estudiantes.

1.1.2 Objetivos

Desarrollar e implementar un sistema que sea capaz de gestionar la información de calificaciones y revalidaciones de los laboratorios pertenecientes al Departamento de Ingeniería en Computación obteniendo la información de una base de datos eficiente para ser puesta a disposición oportunamente a los profesores de teoría y así éstos lleven a cabo su evaluación.

Optimizar el proceso de asignación de calificaciones mediante un portal Web y automatizar el relacionado al tratamiento y distribución de las mismas incluidas las correspondientes a las revalidaciones mediante un sistema que estará alojado en los servidores de la Unidad de Servicios de Cómputo Administrativo (USECAD).

1.1.3 Alcances

Al concluir el proyecto conjunto entre el Departamento de Ingeniería en Computación y la USECAD se conseguirá dar respuesta a las problemáticas mencionadas a través de un sistema que cubrirá las siguientes expectativas:

- Integral.

El sistema estará conformado por diferentes módulos que atiendan las necesidades del Departamento en cuanto al manejo de las calificaciones y garanticen su funcionamiento y eficacia.

- Seguro.

Con la finalidad de reducir las vulnerabilidades del sistema y que éste sea resistente a amenazas, serán implementadas las características señaladas por la arquitectura de seguridad OSI que contempla 5 categorías de servicios de seguridad:

1. *Confidencialidad*. Busca garantizar que la información pueda ser accedida por las partes autorizadas, previniendo su divulgación al viajar por la red por lo que se recurre a la ocultación de información mediante herramientas como la criptografía.

2. *Autenticación.* Sirve para verificar la identidad del usuario a partir de un dato conocido como una contraseña, que al ser validada por la parte con quien desea identificarse demuestra ser quien dice ser. La validación puede darse también a partir de un objeto físico como una llave o por características inherentes al individuo como su patrón de voz. Por lo anterior está estrechamente relacionada con los servicios de Control de Acceso.

3. *Integridad.* Garantiza que la información es recibida exactamente como es enviada protegiendo al sistema contra alteraciones, modificaciones, borrado o inserción de los datos. Cuando algunas de estas acciones ocurren el servicio genera un aviso y utiliza mecanismos para la recuperación de los datos.

4. *Control de Acceso.* Una vez que el usuario ha pasado por el servicio de autenticación tiene acceso al sistema y a sus recursos bajo ciertas condiciones que son determinadas por los privilegios de cada usuario, los atributos de la información y las acciones de escritura, lectura y ejecución que le sean permitidas.

5. *No repudio.* Evita que tanto emisor como receptor nieguen su participación en la comunicación verificando tanto que el mensaje fuera enviado por la entidad especificada como que fuera recibido por la otra parte, empleando generalmente herramientas como las firmas digitales.

- Fiable.

Desde que el alumno sin derecho a reinscripción en laboratorio se registre y que el profesor de laboratorio asigne calificaciones hasta que el profesor de teoría las descargue, todo esto vía un portal Web será un proceso en tiempo real, sin errores y ejecutado por el sistema con lo que se podrá confiar en la veracidad de la información.

- Multiplataforma.

Una ventaja importante que presentará el sistema será el correr en diversas plataformas operativas gracias a las herramientas sobre las cuales estará desarrollado.

- Amigable.

Las interfaces tanto en los portales como con el propio sistema serán amigables con los diferentes usuarios, que bien podrán descargar formatos como el de ASDRI, asignar y descargar calificaciones vía Web, realizar modificaciones y respaldos de una forma sencilla e intuitiva.

1.2 Antecedentes

La Facultad de Ingeniería forma parte del conjunto de Facultades, Institutos y escuelas de la Universidad Nacional Autónoma de México. Nace del interés del gobierno virreinal por resolver los problemas resultantes de la explotación irracional de los recursos mineros y ve su evolución a través de la historia en el Real Seminario de Minería, el Colegio de Minería, la Escuela Nacional de Ingenieros y finalmente con el rango de Facultad en el año de 1959 gracias a la iniciativa de Javier Barros Sierra.

Se organiza en Divisiones que coordinan las Ciencias Básicas y todas las carreras que se imparten, en Secretarías orientadas al desarrollo, la investigación y la administración de actividades y servicios (figura 1.1)

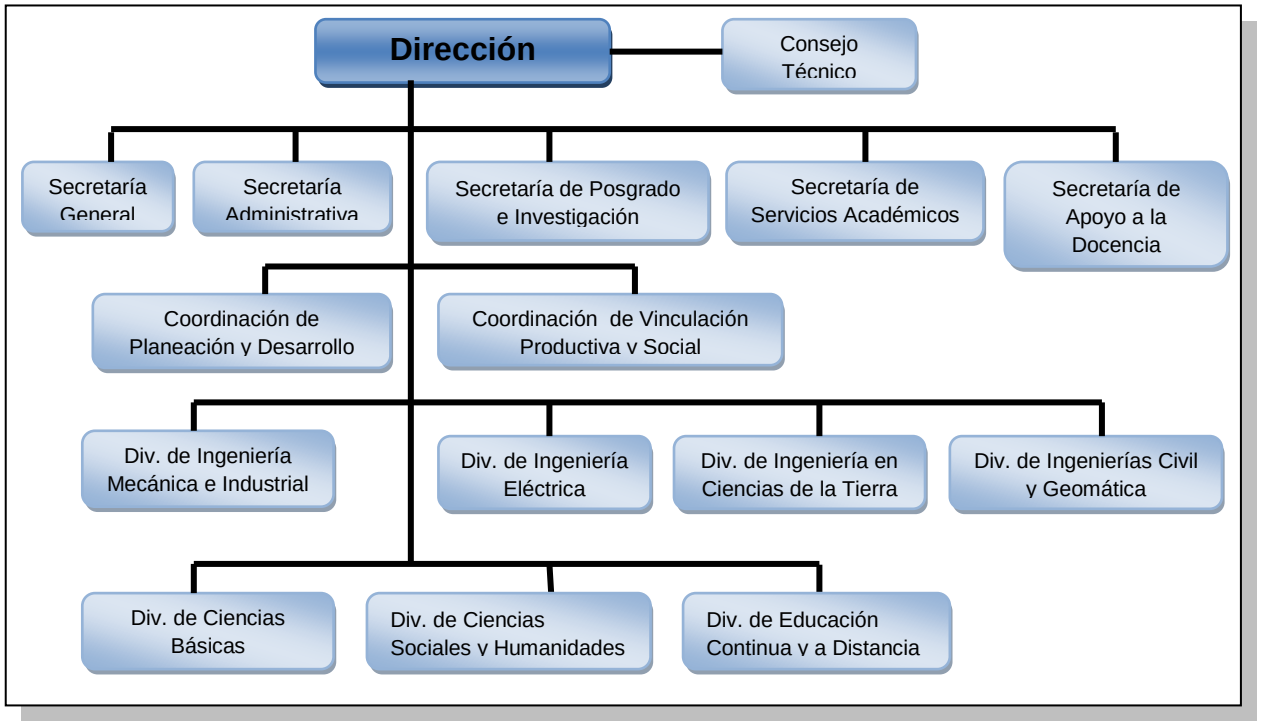


Figura 1.1 Organigrama de la Facultad de Ingeniería

Misión.

Formar integralmente ingenieros competitivos, comprometidos con su país, con valores, habilidades y actitudes que se desempeñen profesionalmente y capaces de orientarse a la investigación y la docencia.

Fomentar la investigación y la difusión de la cultura.

Visión.

Ser la institución al frente de la formación de ingenieros actualizados mediante las opciones de educación continua y a distancia, además de fuente de nuevos conocimientos fruto de la investigación con un impacto óptimo en la sociedad, la cultura y la ecología.

Actualmente ofrece formación en las carreras de:

- Ing. Civil
- Ing. Geomática
- Ing. Geofísica
- Ing. Geológica
- Ing. de Minas y Metalurgia
- Ing. Petrolera
- Ing. Eléctrica y Electrónica
- Ing. en Computación
- Ing. en Telecomunicaciones
- Ing. Mecánica
- Ing. Industrial
- Ing. Mecatrónica

En 2011 se reportó que la carrera de Ingeniería en Computación tuvo una matrícula de 423 alumnos de primer ingreso y 1887 de reingreso de acuerdo a estadísticas de la Facultad de Ingeniería lo que la convierte en la carrera con mayor número de estudiantes con 2310 de un total de 11715 que abarca todas las carreras de la Facultad, apenas seguida por Ingeniería Eléctrica Electrónica e Ingeniería Civil, con 1759 y 1559 respectivamente[1].

Brindar servicios a tal número de estudiantes se vuelve una tarea compleja que es solventada por diversas entidades y Departamentos, tales servicios cubren los rubros académico, administrativo, cómputo y la investigación. Uno de los aspectos más importantes es el proceso de reinscripción que se lleva a cabo vía Internet y del cual se encarga la USECAD semestre a semestre, así mismo el de programación de horarios y administración de calificaciones y revalidaciones de laboratorios que es responsabilidad del Departamento de Ingeniería en Computación, entre muchos otros servicios imprescindibles para el funcionamiento del sistema educativo.

[1].: UNAM :: Facultad de Ingeniería :. Consultada en Abril de 2011 en: http://www.ingenieria.unam.mx/paginas/estadisticas/matricula.php

1.2.1 Departamento de Ingeniería en Computación

El Departamento de Ingeniería en Computación conforma junto con otros seis Departamentos la División de Ingeniería Eléctrica de la Facultad de Ingeniería de la UNAM como puede observarse en la figura 1.2.

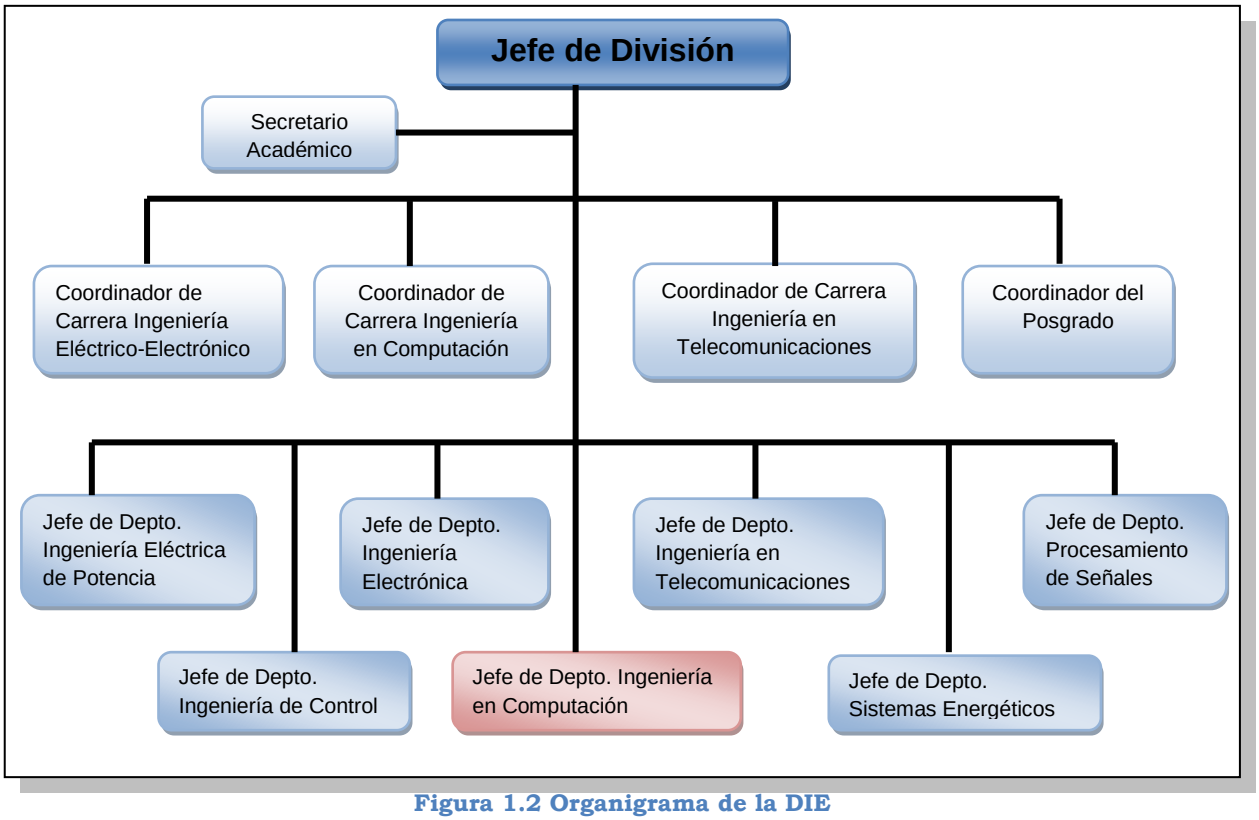


Figura 1.2 Organigrama de la DIE

Misión.

Coordinar todos los aspectos relacionados con la carrera de Ingeniería en Computación.

Proponer los cambios que requiera el Plan de Estudios.

Formar estudiantes de alto nivel en todas las áreas de la computación brindándoles las herramientas necesarias para que sean competitivos en el ámbito profesional tanto nacional e internacional y puedan aplicar sus conocimientos al desarrollo de sistemas complejos que resuelvan problemas y simplifiquen las diversas actividades del quehacer humano.

Favorecer la especialización de alumnos y docentes en campos de la ingeniería de hardware y software, redes y seguridad, bases de datos, sistemas inteligentes y computación gráfica e ingeniería biomédica, preservando el interés por mantenerse actualizado y a la vanguardia.

Visión.

Fomentar la Investigación y la Docencia.

Promover la vinculación con empresas y otras instituciones educativas nacionales e internacionales.

Algunas funciones del Departamento de Ingeniería en Computación.

- Mantenimiento y actualización de la agenda del personal académico.
- Mantener al día la base de horas/profesor y horas/ayudante con las que cuenta.
- Organización, planeación y realización de horarios de cada semestre.
- Elaboración de los movimientos (contrataciones, bajas, altas, prórrogas, etc.) del personal académico y sus justificaciones.
- Revalidación de calificaciones de los laboratorios.
- Recopilación de calificaciones de los laboratorios que pertenecen al Departamento, así como la distribución de calificaciones de éstas hacia los profesores de teoría.

1.2.2 Unidad de Servicios de Cómputo Administrativo

La Unidad de Servicios de Cómputo Administrativo (USECAD) forma parte de la Secretaría de Servicios Académicos (SSA) que es el organismo de la Facultad de Ingeniería encargado de proporcionar los servicios de cómputo para las actividades de administración escolar. A continuación se muestra en la figura 1.3 el organigrama de la SSA.

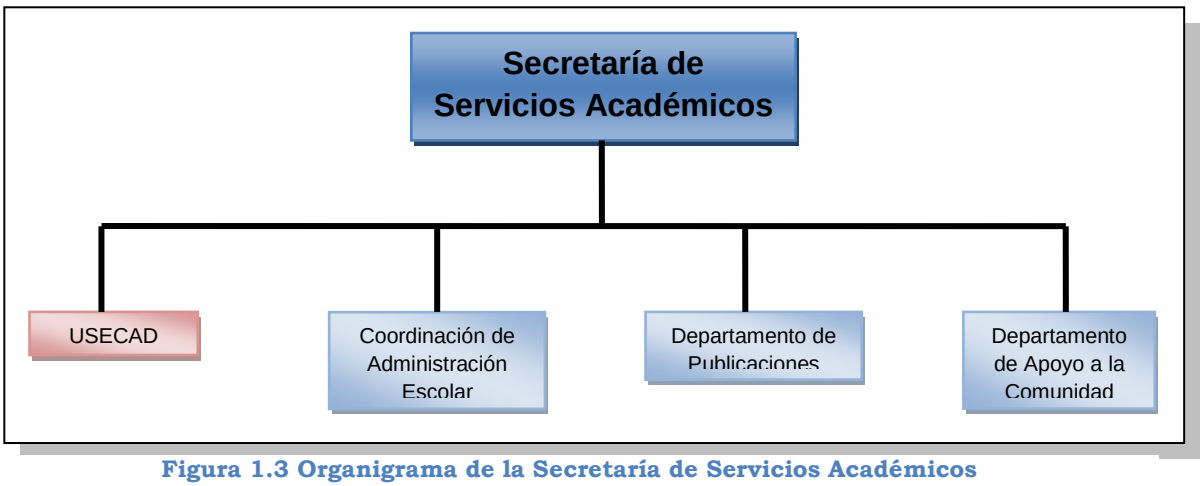


Figura 1.3 Organigrama de la Secretaría de Servicios Académicos

Misión.

Brindar los servicios de cómputo, soporte técnico, desarrollo de sistemas y consulta a su base de datos tanto alumnos como a los funcionarios de la Facultad de Ingeniería.

Visión.

Actualizar la infraestructura y los sistemas que mantienen los procesos académicos y administrativos para mejorar la calidad de servicios a los usuarios.

Algunas funciones de la USECAD.

- Administrar la red de cómputo que permite a los funcionarios de la Facultad el acceso a las bases de datos de inscripciones, información de profesores y alumnos, horarios e historiales.

- Inscripción y reinscripción de los alumnos vía Web.

- Proporcionar servicios de consulta vía Web sobre horarios, cupos, vacantes y resultados de la inscripción.

- Actualizar los recursos de los servidores de bases de datos y el equipo de cómputo de la SSA para asegurar su correcto funcionamiento.

1.3 Marco de referencia

Conceptos básicos de programación

Un programa es lo que marca la diferencia entre una máquina de escribir y una computadora –sin hacer de lado la riqueza del hardware- dado que permite recrear y resolver situaciones al familiarizarlas con metodologías y lenguaje matemático atendiendo a cada variable con una acción.

Un **programa** es un conjunto de instrucciones que guían a la computadora y sus recursos para resolver algún problema, que interactúan con el usuario y los datos que procesa.

Mediante un programa se puede representar cualquier situación de la realidad, simular fenómenos y realizar cálculos extensos y complejos.

El proceso de un programa implica varias interrogantes tales como:

- ¿Para qué servirá?
- ¿Qué datos usará?
- ¿Qué resultados producirá?
- ¿Cómo se operaran los datos para obtener lo esperado?

A partir de las respuestas y teniendo en mente la estructura interna y funcionamiento más convenientes se procede a diseñarlo con ayuda de alguna representación simbólica o textual resultando en un algoritmo que posteriormente se codificará con un lenguaje de programación.

Este proceso consta de un conjunto de instrucciones legibles por un individuo que reciben el nombre de código fuente, pero al ser sólo palabras y símbolos no son identificados por la computadora ya que ésta sólo es capaz de procesar elementos binarios, es decir, una serie de ceros y unos.

Tal secuencia que puede ser de millones conocida como lenguaje máquina es el resultado de traducir el código fuente mediante un compilador[2], el cual será

[2] Compilador. Programa que lee el código fuente y lo divide en instrucciones, estructuras de control, operadores, etc. propias del lenguaje de programación validando que sea correcto sintáctica y semánticamente, posteriormente lo traduce a código intermedio o código máquina que ya es ejecutable por el equipo.

elegido de acuerdo a la plataforma operativa donde será ejecutado el programa en base a las necesidades, finalidad y complejidad del mismo.

Sin embargo el mayor peso en la creación del programa recae en el razonamiento del desarrollador, en su forma de analizar, diseñar y aprovechar los recursos a su alcance para ofrecer soluciones considerando en el proceso cada detalle y cada posible falla.

Una aptitud importante es también cómo se comunica con la computadora, tener en mente que hablarle es como instruir a un niño pequeño en cierta actividad, a quien deben darse instrucciones precisas en un lenguaje que comprenda y recordar que tanto sus experiencias como su poca capacidad debidas a su corta edad le impiden la toma de decisiones ante algún tropiezo dado que sólo seguirá el camino que se le haya trazado.

En conclusión la computadora será el perfecto asistente que ejecute todo lo que se le indique pero no podrá afrontar problemas por sí misma a menos que el programador los haya contemplado y le enseñe una posible solución como parte de las instrucciones que reciba en cierto lenguaje de programación, que si bien existen los de alto nivel distan mucho del natural.

Los lenguajes de programación se pueden clasificar de acuerdo a la figura 1.4:

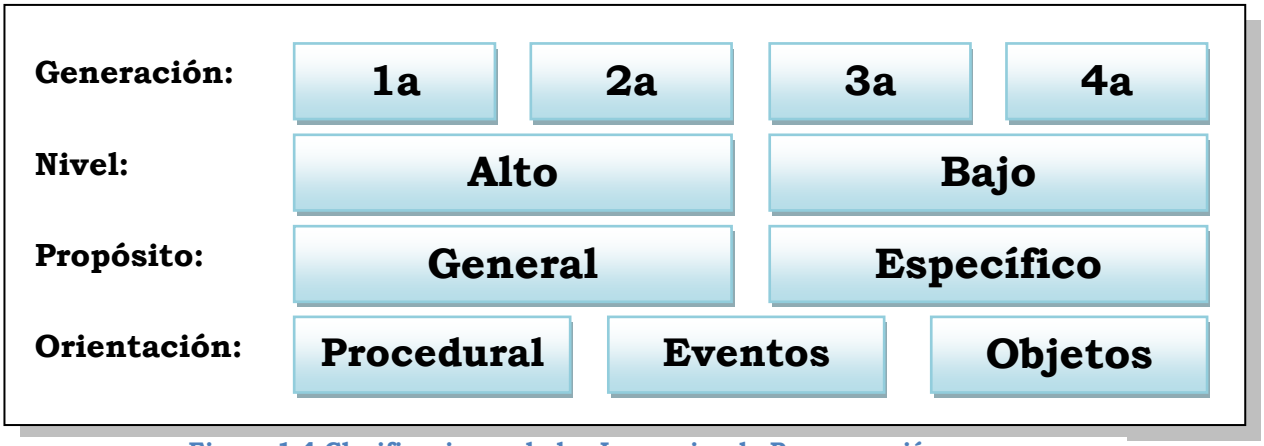


Figura 1.4 Clasificaciones de los Lenguajes de Programación

La *generación* refiere a que tan compleja y numerosa sea la secuencia de números binarios representada por cada símbolo que compone el lenguaje.

El *nivel* comprende el número de plataformas operativas en las que el programa desarrollado en el lenguaje podrá ejecutarse.

El *propósito* contempla las áreas del conocimiento humano hacia las que está destinada la aplicación y sus limitantes, como puede ser una actividad específica como la graficación hasta la apertura del lenguaje para desarrollar aplicaciones de bases de datos, científicas, etc.

La *orientación* está asociada a la forma en que las instrucciones que conforman el código fuente se organizan y se estructuran internamente, así como la forma en que se ejecutarán.

El caso de la orientación a objetos destaca por su creciente popularidad en la actualidad debida a sus promesas de reutilización de código, de construir aplicaciones modulares, distribuidas y fácilmente interconectables con un mayor nivel de abstracción[3].

Tiene un marcado contraste con lenguajes no orientados a objetos como C y Basic de tipo procedural y que basan su estructura en el concepto de procedimiento o función.

La función deriva de la filosofía de seccionar un problema en subproblemas más simples y que se enfocan a una tarea específica a través de un bloque de instrucciones que operan sobre los datos que recibe llamados argumentos y produce un resultado. Así el programa se integra por una sucesión de funciones y las llamadas entre ellas que pueden estar incorporadas en el lenguaje, en el sistema operativo o creadas por el usuario.

La programación estructurada busca implantar un procedimiento como solución a la cuestión: ¿cómo se resolverá el problema?, y trata de guiar secuencialmente la estructura del código. Mientras que la programación orientada a objetos (POO) no se preocupa por las instrucciones ni su orden, sino por el quién efectuará el procedimiento al abstraer la naturaleza de las cosas y la conexión que hay entre ellas.

[3] La abstracción es la capacidad de notar sólo lo esencial, la acción misma sin detenerse en los detalles, de enfocarse en el ¿qué hace? y no en el ¿cómo lo hace?

1.3.1 Programación orientada a objetos

Conceptos básicos

La POO ofrece la ventaja de representar entidades de la realidad sobre las cuales realizar operaciones y provocar comportamientos. Tales entidades poseen características propias que las distinguen, acciones que las manipulan y tienen su equivalente en la computación en los objetos.

Un programa orientado a objetos se constituye por un conjunto de objetos que interactúan entre si y cuya funcionalidad, entradas y salidas permiten la resolución de un problema como se observa en la figura 1.5.

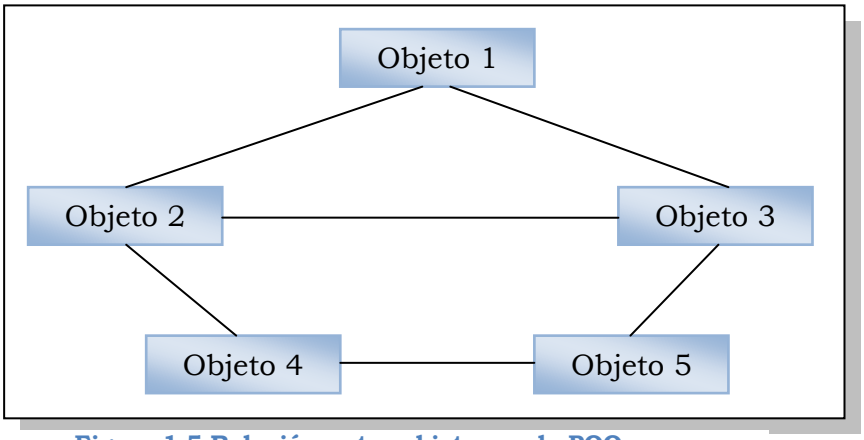


Figura 1.5 Relación entre objetos en la POO

Un objeto es la representación en un programa de un concepto y contiene toda la información necesaria para abstraerlo: datos que describen sus atributos y operaciones que pueden realizarse sobre los mismos.[4]

Los **atributos** también conocidos como datos expresan la información que se conoce del objeto, su forma, apariencia y características a través de valores.

Los **métodos** son comportamientos equivalentes a las funciones en los lenguajes procedurales que representan las acciones que el objeto podrá realizar al manipular sus atributos.

[4] CUENCA, Pedro, *Programación en Java,* p. 272.

A pesar de ser un estilo muy diferente de programar, la POO sigue también una metodología que inicia en el planteamiento del problema a partir de los datos de entrada, el proceso y la salida esperada; identificar los objetos como sustantivos; el diseño del algoritmo en base al diagrama de clases y las pruebas de escritorio.

Una vez explicado el concepto de objeto será más sencillo explicar el de clase que es fundamental en el diseño porque con una basta para representar objetos iguales que hacen lo mismo, es decir, con características y comportamientos comunes.

La **clase** es una representación abstracta y formal que describe un conjunto de objetos del mismo tipo. Puede verse como un molde de helado o gelatina la cual posee atributos definidos como forma, tamaño y sabor que sirve para crear muestras que si bien poseen el mismo tipo de atributos, sus valores pueden cambiar como el caso del sabor como se observa en la figura 1.6.

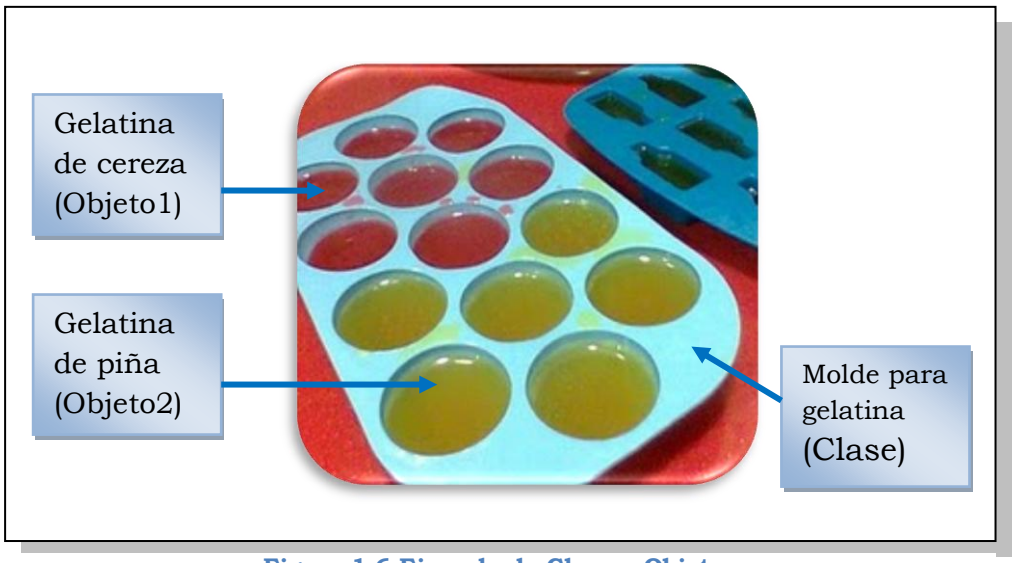


Figura 1.6 Ejemplo de Clase y Objetos

En términos de computación, la clase será la plantilla que define los atributos y métodos (llamados miembros de clase) comunes que serán copiados a cada objeto (llamado instancia) creado a partir de ella pero que tendrá valores propios.

Por ejemplo en la figura 1.7 se observa un diagrama de la Clase Empleado con atributos y métodos definidos, de ésta se crean dos instancias con valores distintos que representan a dos empleados.

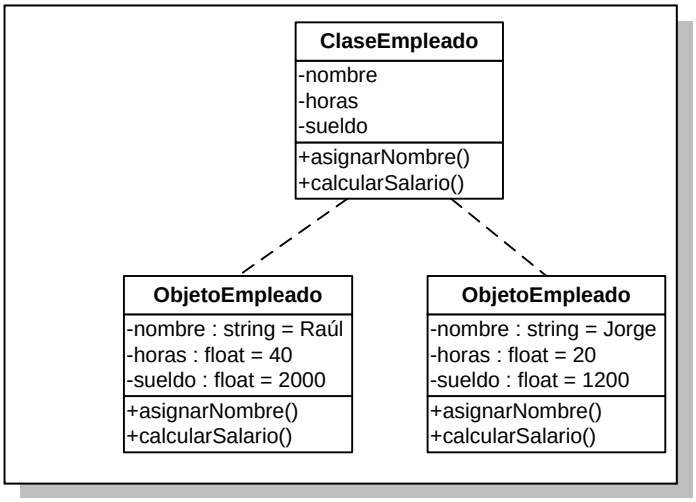


Figura 1.7 Diagrama de Clase y Objetos

Las clases constituyen la unidad mínima modular que permiten crear instancias, nuevos tipos de datos y estructurar código. Se diseñan teniendo en mente el código y los datos de forma conjunta, imaginando si son leídos o generados, así como los métodos que se encargarán de acceder a ellos.

Los objetos no están aislados, hacen uso de métodos para comunicarse con otro objeto pidiéndole que realice cierta acción y con los datos que contiene a través de mensajes.

Para que una aplicación realmente esté desarrollada orientada a objetos y no sólo basada en ellos debe cumplir con tres condiciones: encapsulamiento, herencia y polimorfismo.

El **encapsulamiento** permite un control total sobre "quién" o "qué" puede acceder a los miembros de un objeto haciendo invisible su estructura interna y protegiendo su información frente a accesos sin autorización.

A pesar de que el objeto adquiere la cualidad de caja negra suele presentar métodos de acceso para que otros objetos consulten o modifiquen sus atributos de una forma segura sin la necesidad de dar a conocer como se integra ni su comportamiento dentro de sí mismo. Observe la figura 1.8 donde la clase empleado posee miembros de tipo privado (su nombre está precedido por signo

"-") invisible para otras y de tipo público (su nombre está precedido por signo "+") a la vista y disposición del resto.

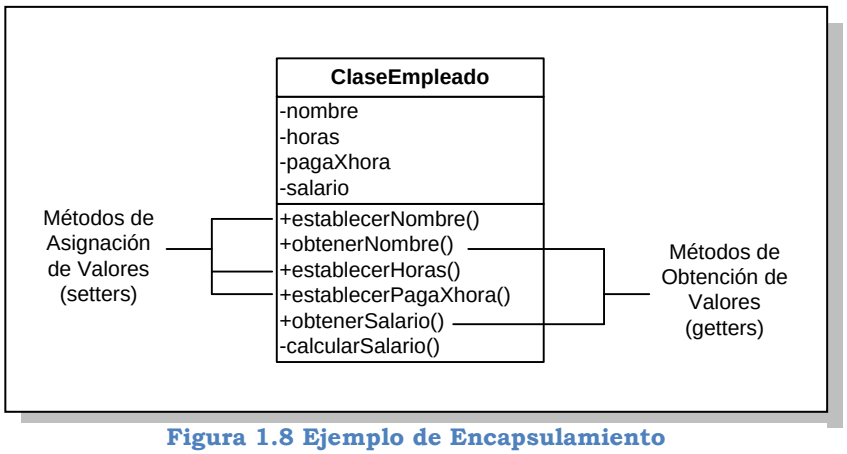


Figura 1.8 Ejemplo de Encapsulamiento

El encapsulado reduce la complejidad al usar un objeto dado que sólo se emplea su interfaz de comunicación sin prestar atención a su implementación. Se distinguen dos tipos de interfaces esenciales, la pública constituida por los métodos que son conocidos y sirven de enlace con otros objetos; y la interfaz privada que incluye lo necesario para funcionar y comunicarse internamente siendo visible sólo por objetos de la misma clase o incluso únicamente por el objeto mismo. Cualquier intento de otro objeto por acceder a la parte privada arrojará un error del compilador que garantiza el cumplimiento de los modificadores de acceso que además de ser público y privado, también se usan de tipo protegido, estático o abstracto.

El ser humano abstrae los sistemas complejos de forma jerárquica dividiéndolos sucesivamente en sistemas más simples hasta conseguir su unidad mínima. Puede verse de forma similar a un árbol genealógico identificando primero al objeto más genérico de la especie y después a sus derivados.

La **herencia** se basa en este principio para organizar jerárquicamente las clases de un programa, donde una clase puede crearse a partir de una existente llamada clase padre o superclase y tomar todos sus atributos y métodos.

No obstante, la clase derivada o hija puede redefinir los miembros heredados o añadir nuevos que la especializan en cierta tarea y delimitan su accionar. Así se logra simplificar diseños y reutilizar código en común.

La clase padre debe diseñarse con cuidado pues sus miembros afectarán a todas las clases herederas, por lo cual si un error es encontrado o se pretende redefinir un método basta con corregirlo en la clase padre y los cambios se verán reflejados inmediatamente en sus descendientes.

La clasificación de jerarquías con la herencia es una tarea compleja dado que deben establecerse las relaciones y agrupaciones más convenientes, seleccionando los métodos y atributos más genéricos y comunes a todas las clases hijas lo que convierte al diseño en una actividad de prueba y error. Una técnica que ayuda a identificar está relación se basa en la expresión "es un" que se observa en "automóvil es un vehículo" pero sin caer en la expresión "tiene un" como por ejemplo "Juan tiene un automóvil".

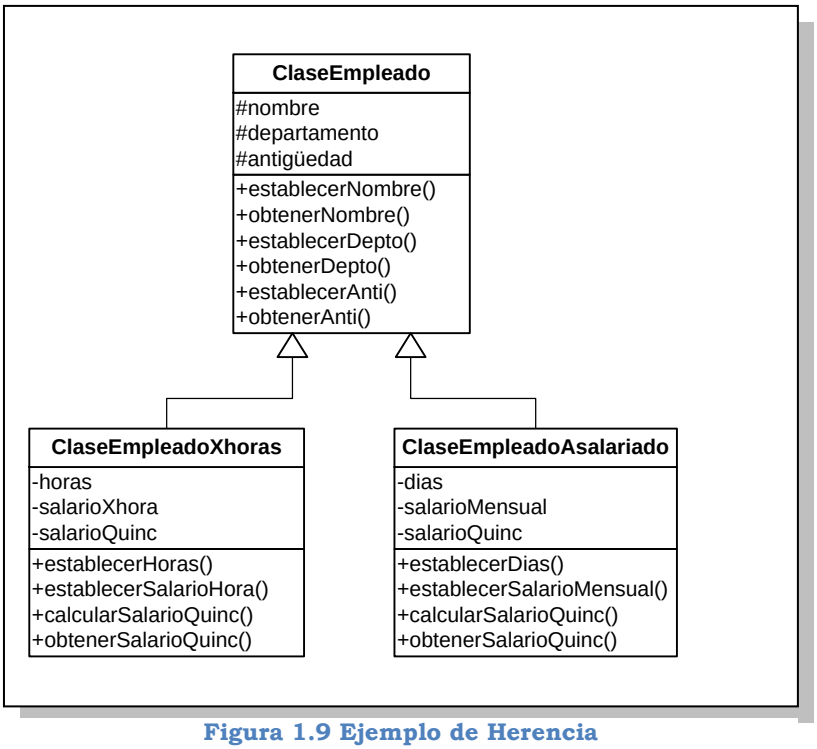


Figura 1.9 Ejemplo de Herencia

Al retomar la clase genérica Empleado, pueden extenderse dos clases que hereden sus miembros pero se especialicen en dos tipos de empleado diferente que poseen atributos y métodos propios. Por un lado se tiene el empleado que trabaja por horas cuyo contexto se define por el número de horas trabajadas y el salario correspondiente por fracción de tiempo; por otro se tiene al empleado asalariado con un salario mensual fijo como se observa en la figura 1.9.

El **polimorfismo** es la propiedad que permite al objeto modificar su forma y adaptar su comportamiento en función del ambiente y la exigencia. Se observa cuando un método realiza la misma operación sin importar el número de argumentos que recibe o cuando éstos son de diferente tipo de dato, tal caso recibe el nombre de Overloaded o sobrecarga. Otro caso de polimorfismo es el de Overridden o sobreescritura que consiste en reemplazar un atributo o método heredado por el que realmente necesita la clase, es decir, la implementación del código cambia de acuerdo a la nueva clase pero a fin de respetar el encapsulamiento es necesario respetar el número de argumentos y tipos de datos.

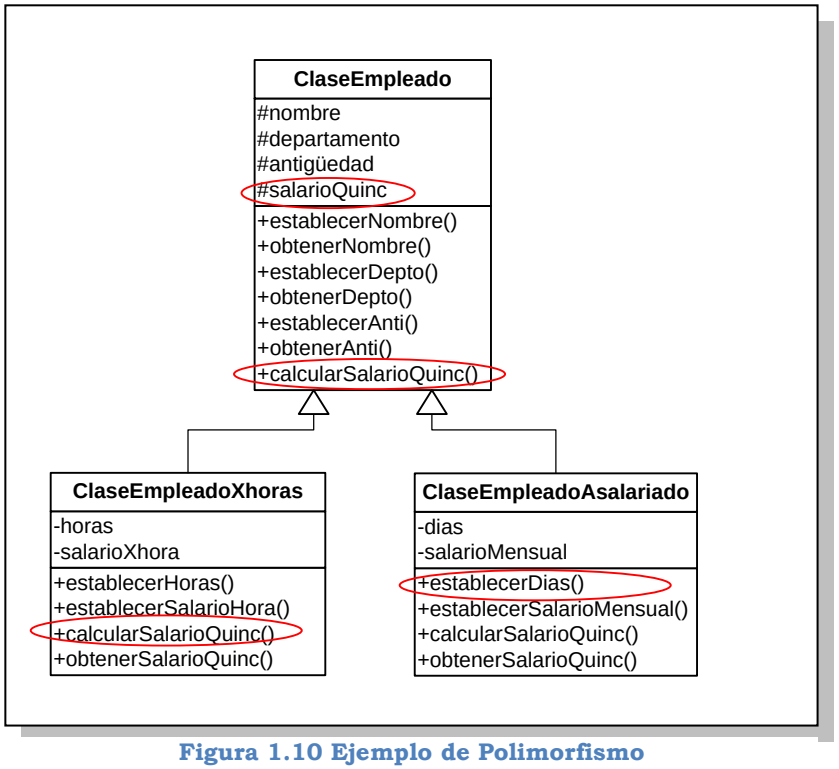


Figura 1.10 Ejemplo de Polimorfismo

En la figura 1.10 se observa el uso del polimorfismo sobre la clase Empleado, que se ve modificada al incluir el atributo salarioQuinc y el método calcularSalarioQuinc que suben de nivel para poder ser heredados. Las clases hijas modifican este método de acuerdo a sus especificaciones, mientras que en la ClaseEmpleadoXhoras el cálculo utiliza las *horas* trabajadas por el *salarioXhora*; en la ClaseEmpleadoAsalariado el cálculo resulta del *salarioQuinc* fijo en función de los días trabajados. En conclusión el método lleva el mismo nombre, hace la misma función pero el mecanismo y los datos que emplea cambian.

Ventajas:

- Abordar la resolución de un problema de forma metódica que lleva de lo general a lo particular.
- El diseño tiende a reflejar fielmente las condiciones del problema dado por el mayor nivel de abstracción.
- Reutilización del código que reduce la extensión del programa.
- Las clases pueden desarrollarse de forma separada dado que no influyen en las demás sólo tomando en cuenta sus interfaces, hasta las dispuestas por otros programadores en Internet.
- Mejora el mantenimiento del software y su flexibilidad al preservar la interfaz de una clase a pesar de re-implementarse.
- Modificadores de visibilidad para ofrecer seguridad y condiciones de acceso a las clases.

Desventajas:

- Requiere mucho mayor planeación, buen diseño y técnica de programación.
- La abstracción y la jerarquía de clases no son tareas triviales.
- El análisis del problema es más complejo dado que deben identificarse todos los elementos que componen las clases y la relación entre ellos.
- El tiempo de ejecución presenta un leve retardo ya que los métodos usados en la herencia llevan más tiempo que la llamada a una función de lenguaje estructurado.

1.3.2 Lenguaje de programación Java

¿Qué es Java?

Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems cuya sintaxis está basada en C++ pero elimina todos los elementos causantes de error como el manejo de apuntadores.

Tiene su origen en el mercado de la electrónica de consumo y los electrodomésticos cuyos reducidos recursos de cómputo y memoria requerían un lenguaje sencillo y de poco tamaño.

Los lenguajes como C y C++ se enfrentaban además al surgimiento de nuevos chips de menor costo pero con arquitecturas diferentes lo que conducía a re-compilar el software frecuentemente.

Java pretende resolver el problema de arquitecturas incompatibles tanto en equipos de cómputo, sistemas operativos, sistemas de ventanas y dispositivos electrónicos, es decir, que la aplicación corra sobre cualquier plataforma de hardware y software, pero además brindar soporte para aplicaciones en red y distribuidas.

Sin embargo, el ámbito de la Web fue quien lanzó a Java al éxito al dotarlo del dinamismo del que carecían los documentos Web escritos en lenguaje HTML que constaban originalmente de texto, imágenes y vínculos a otros documentos.

Posteriormente evolucionó para incluir contenido dinámico como audio y video, no obstante era necesario contar con plug-ins en el navegador para interpretarlo o en su defecto descargarlo y así usar el programa apropiado instalado en el equipo.

Con el auge del contenido interactivo se evidenció la necesidad de un lenguaje de programación con las facultades para incluir en las páginas Web interfaces de usuario y conectar los datos recibidos en formularios con los repositorios necesarios.

Surge Java para sustituir a los limitados lenguajes, proporcionar a los navegadores la capacidad de ejecutar programas y reproducir contenido multimedia y desarrollar todo tipo de aplicaciones.

Tipos de aplicaciones en Java

- *Applets.*

Son mini aplicaciones que se ejecutan dentro de un navegador al cargar una página HTML desde un servidor Web y que no requieren instalación.

El navegador puede interpretarlos dado que se incrustan en la página con una etiqueta al igual que una animación de flash.

Mediante los applets se logra también integrar sonidos, video y contenido 3D.

- *Aplicaciones de Consola.*

Programas independientes que se desarrollan al igual que los lenguajes tradicionales pero limitados a sólo texto en una línea de comandos como en el ambiente de MS-DOS.

- *Aplicaciones Gráficas.*

Utilizan las clases destinadas para gráficos como AWT y Swing para brindar una amigable interfaz de usuario a partir de un entorno visual sencillo con ventanas, imágenes y botones de acción.

- *Servlets.*

Programas sin interfaz gráfica que se ejecuta en el servidor Web y devuelve una página HTML.

¿En qué se usa?

La versatilidad, portabilidad y potencia de Java han colocado aplicaciones en diversos ámbitos de la vida común y en las tecnologías de la información, se encuentran en consolas de videojuegos, menús de películas en alta definición, integración de juegos y medios en teléfonos móviles, navegadores y servicios Web, procesamiento de datos de foros en línea y tiendas, comercio electrónico, facilitando la carga de imágenes en redes sociales, aplicaciones de escritorio y empresariales, chat, juegos en línea, dispositivos electrónicos como impresoras y cámaras, etc.

Proceso de compilación e interpretado

La independencia de la Plataforma y eficacia en la diversidad de aplicaciones se deben a que Java realiza tanto compilación como interpretación de código lo que convierte al ambiente de desarrollo un tanto particular como se observa en la figura 1.11. Para programar en Java se requiere:

- *Entorno de desarrollo.*

Sun distribuye gratuitamente el JDK (Java Development Kit) que incluye un conjunto de programas y librerías necesarios para codificar, compilar y ejecutar los programas en Java.

- *Editor de programas.*

La codificación puede escribirse en cualquier editor de texto plano o hacer uso de herramientas de desarrollo de tipo IDE[5] como JBuilder, NetBeans, JCreator, Eclipse y Visual Studio.

- *Compilador.*

En el JDK el compilador se llama javac.exe y está encargado de realizar un análisis de sintaxis del código fuente con extensión *.java devolviendo después de la compilación archivos con extensión *.class con código intermedio que recibe el nombre de ByteCode.

- *Biblioteca de clases.*

Un programa en Java no parte de cero, es necesario invocar las clases necesarias que le otorgan cierta funcionalidad ya desarrollada y que vienen predefinidas en la biblioteca de clases o comúnmente llamada API (Application Programming Interface). Al instalar el entorno de desarrollo quedan disponibles en la carpeta include dentro de jdk y están organizadas en una jerarquía de clases a través de paquetes contenidos en archivos con extensión *.H. Además

[5] IDE: Entorno Integrado de Desarrollo que provee un marco de trabajo amigable para uno o más lenguajes de programación al incorporar en la misma aplicación un editor de código, un compilador, un depurador o debugger, un constructor de interfaz gráfica y ambiente de ejecución para evitar cambiar de aplicaciones.

de éstas puede hacerse uso de clases comerciales o desarrolladas por otros programadores.

- *Máquina Virtual de Java (JVM).*

El código intermedio resultante de la compilación no es código ejecutable en ninguna plataforma de hardware pues no corresponde con las especificaciones de procesadores reales, pero si con una máquina virtual de carácter puramente teórico con set de instrucciones, tamaño en bits y tipos de datos bien definidos. La llamada Máquina Virtual Java (también conocida como JRE o entorno de ejecución) se encarga de interpretar el código intermedio pasándolo a código máquina del procesador donde se está ejecutando. En este punto es donde se consigue la portabilidad dado que una vez compilado el código fuente y obtenido el código intermedio ya no es necesario modificarlo para cambiar de plataforma, sólo se requiere que posea el entorno de ejecución adecuado que ya es dependiente de la máquina.

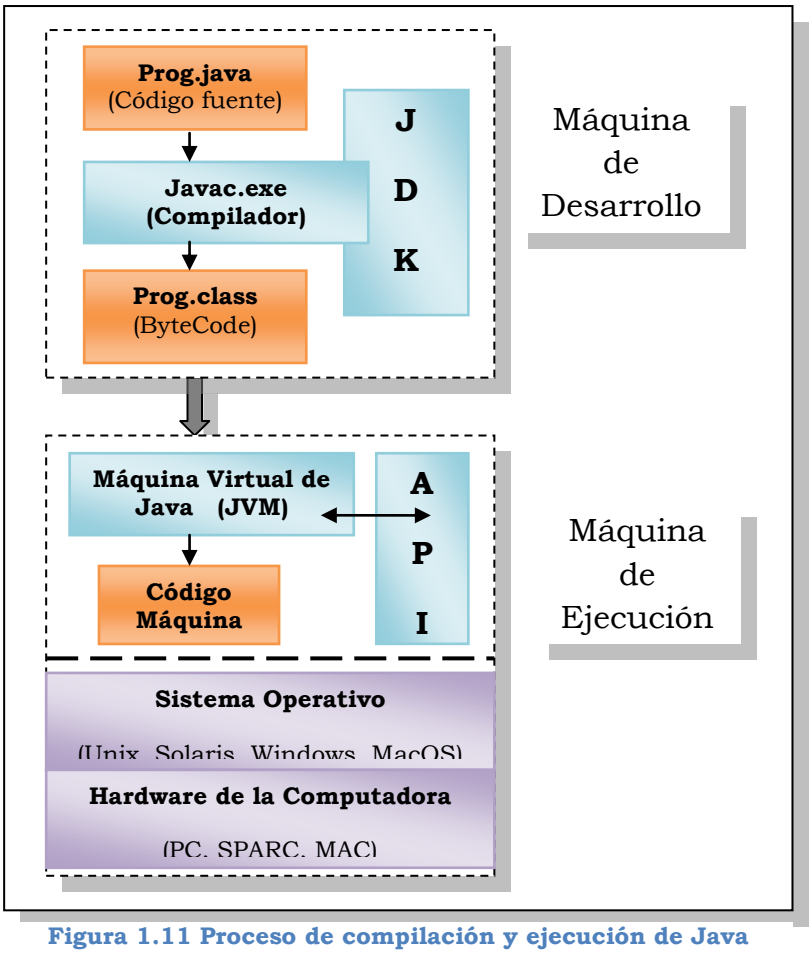


Figura 1.11 Proceso de compilación y ejecución de Java

Características

- *Sencillo.*

Aunque recoge sólo las características esenciales de C y C++ puede abordar cualquier tipo de problema, además es de rápida adopción y permite construir aplicaciones ligeras que no consumen muchos recursos.

Reduce los errores al eliminar el uso de apuntadores, registros, macros y la necesidad de liberar memoria y el tiempo perdido derivado de esto puede destinarse al diseño del programa.

- *Orientado a Objetos.*

A comparación de los programas en C que ejecutan sus instrucciones de forma secuencial y de C++, que a pesar de estar pensado como extensión de C para emplear entidades independientes pueden no usarse, Java es un lenguaje que basa plenamente su estructura interna y funcionalidad en los objetos y la comunicación entre ellos. Además soporta las características esenciales de esta orientación como la herencia, encapsulación y polimorfismo.

- *Distribuido.*

Las aplicaciones Java pueden ajustarse a la arquitectura cliente – servidor, donde el programa puede actuar como servidor permitiendo el enlace con otros o bien con bases de datos, mientras que el usuario accede a estos servicios mediante una interfaz que puede ser brindada por un applet.

Con esto se evita la instalación de software y la descarga de actualizaciones ampliando su alcance dado que la corrección de errores y mejoras sólo se realizan del lado del servidor pasando inadvertidas para el usuario.

Además de soportar los servidores más populares, las librerías estándar de Java favorecen el desarrollo de aplicaciones distribuidas donde los servicios pueden estar alojados en diferentes sitios pero comunicados entre sí por la red.

- *Robusto.*

La robustez de un programa radica en su resistencia a errores y su capacidad para manejarlos con el fin de evitar un alto en la ejecución de forma repentina.

Por tanto deben realizarse verificaciones tanto en tiempo de compilación como de ejecución.

Entre las singularidades de Java es que obliga a la comprobación de tipos y la declaración explícita de métodos, además realiza el manejo de la memoria, la comprobación de límites de arreglos y la verificación del código intermedio.

Otro aspecto importante es que pueden recogerse los errores que se produzcan en tiempo de ejecución por medio del uso de excepciones que permiten darles posibles soluciones a los que se piensen puedan ocurrir y así garantizar la continuidad del programa.

- *Independiente de la Plataforma.*

Un programa puede ejecutarse sin cambios en cualquier ambiente de hardware y software sin importar en que máquina fue generado, abarcando desde una PC, Mac o servidor hasta dispositivos electrónicos como televisiones o teléfonos móviles. Esto es posible porque el código pre-compilado es independiente de la plataforma por lo cual sólo se requiere el entorno de ejecución o JRE (conocido también como Java Runtime Environment, Runtime, Java Virtual Machine) que se encargará de interpretar las instrucciones y está disponible para los sistemas operativos Solaris, SunOS, Windows, MacOS, Linux.

- *Seguro.*

Java se basa en 3 herramientas para garantizar una de sus mayores fortalezas, la seguridad.

1. *Verificador de código intermedio.*
   Comprueba el correcto comportamiento del código y el cumplimiento de las reglas de Java, detecta fragmentos de éste de carácter ilegal que viole los derechos de acceso o convierta tipos y clases de un objeto.

2. *Verificador de Clases.*
   Proporciona las clases necesarias y se asegura que sean las originales de Java buscando primero entre las locales y después entre las procedentes del exterior, impidiendo así los Caballos de Troya y la suplantación de clases.
   Puede verificar además el origen del código mediante firmas digitales validando sus privilegios antes de instanciar un objeto y cargarlo en memoria.

Por otro lado, en el caso de los applets existe una mayor seguridad dado que los intérpretes de los navegadores Web presentan más restricciones que bloquean la ejecución de aplicaciones nativas de la plataforma, abrir archivos o usar el equipo como "zombie[6]" sin el conocimiento del usuario.

3. *Administrador de Seguridad.*
   Permite al usuario elegir niveles de seguridad para la ejecución de programas y de confianza de acuerdo al sitio de procedencia del código que puede ser identificada por ejemplo mediante firmas digitales.

En conclusión, las medidas de seguridad evitan que los programas accedan directamente a los recursos de la computadora como memoria y disco duro previniendo su interacción con ciertos virus. Además cada operación es vigilada tanto en la compilación como al momento de ser ejecutada por el intérprete.

- *Portable.*

El concepto de portabilidad se centra en la ya mencionada independencia de la plataforma del código compilado que se extiende también a los tipos de datos, los cuales ocupan el mismo espacio a diferencia de C y C++, así por ejemplo un entero siempre será de 32 bits en complemento a 2.

Asimismo se recurre a un sistema abstracto de ventanas aprovechando la semejanza que presentan en cuanto a componentes y funcionalidad en todos los sistemas operativos.

- *Interpretado.*

Java involucra 2 procesos casi antagónicos en la programación para obtener una aplicación, la compilación y la interpretación de código, éste último permite la independencia de la plataforma y el uso de menos recursos pero a costa de una reducida velocidad de ejecución donde en comparación con C es de 10 a 30 veces más lento.

Sin embargo, deben tomarse en cuenta los mecanismos de seguridad mencionados anteriormente y la búsqueda de la mayor optimización durante la compilación los parcialmente causantes de esta lentitud.

[6] Equipo que al ser infectado por algún tipo de malware es usado por otro individuo sin conocimiento del usuario para infectar equipos, distribuir SPAM o generar tráfico con más zombies para atacar un servidor.

- *Multithreading.*

Haciendo alusión a las traducciones más populares multithreading quiere decir multihilo, multitarea o multihilvanado. Los threads por tanto son pequeños procesos independientes que integran uno mayor.

Un programa en Java puede crear varios threads o hilos de ejecución que se ejecutan en paralelo con una tarea específica cada uno, por ejemplo, la recepción de datos del usuario por el teclado, la lectura de archivos, cálculos, la descarga de cierto contenido Web como imágenes mientras se observa el resto de la información sin tener que esperar a que concluya, etc.

Además los threads pueden asignarse a diferentes microprocesadores si se cuenta con sistemas de este tipo. No obstante, el rendimiento multitarea es acorde al sistema operativo y a su capacidad para manejarlo.

- *Dinámico.*

Bajo el supuesto de que un programa en Java no es un ente compacto cada vez que se compila sino un conjunto de módulos que se cargan y se conectan hasta el tiempo de ejecución, es posible actualizar las librerías añadiendo nuevos métodos y atributos a sus clases sin afectar la aplicación actual. En caso de que ésta se ejecute en red serán traídos automáticamente los fragmentos necesarios o que no se entiendan.

- *Difundido.*

Al ser un lenguaje versátil en cuanto a las aplicaciones y áreas de desarrollo abarca el entretenimiento, programas simples y empresariales, teléfonos móviles, etc. Tal alcance lo ha llevado a contar con una vasta documentación, fuentes en Web, foros y libros de gran dominio entre los programadores.

- *Garbage Collector.*

Una carga importante para el programador de C++ es la administración de la memoria de forma manual, donde la falla en su asignación y desalojo puede llevar a fugas de memoria, consumir más de la necesaria o cuelgues del programa.

Java retira esta responsabilidad al programador mediante el recolector de basura que de forma automática e invisible asigna y libera los recursos.

Cuando el programa crea un objeto pide al sistema la memoria necesaria siendo vigilado su ciclo de vida por el entorno (Java Run-Time) a través de un thread de baja prioridad que lo examina para determinar en qué momento se ha dejado de usar y no quedan referencias hacia él, entonces lo borra y libera la memoria que estaba ocupando.

Ventajas:

- El código puede ejecutarse sin cambios en cualquier dispositivo que cuente con el entorno Java respectivo.
- Al ser orientado a objetos permite resolver un problema metódica y modularmente facilitando el desarrollo cooperativo.
- Menos errores al eliminar el manejo de apuntadores y la gestión de la memoria dado que su asignación y liberación se realizan de forma automática.
- Admite el uso de técnicas para optimizar el desempeño y acelerar la ejecución.
- Facilita la inclusión de contenido interactivo en páginas Web sin el uso de paquetes multimedia o la instalación de plug-ins.
- La librería de clases es estándar para todas las plataformas y permite afrontar problemas de cálculos matemáticos, acceso a bases de datos, cómputo gráfico, etc.
- Desarrollo de aplicaciones modulares y distribuidas.

Desventajas:

- La más representativa es la reducida velocidad de ejecución debida a la Máquina Virtual de Java que interpreta constantemente el código intermedio consumiendo buena cantidad de recursos de procesamiento y ofrece un bajo rendimiento a comparación de los lenguajes compilados.
- A pesar de la similitud con C++, Java es un lenguaje por sí mismo con sus propias reglas que deben aprenderse.

Servlets

El uso de la Web como se ha mencionado se enfocaba a la consulta de documentos estáticos mediante el navegador del usuario (que bien puede tomar el nombre de cliente). Tales documentos se alojaban en una computadora que fungía como servidor el cual se limitaba a enviar el archivo *.HTML a través de la red cuando el usuario lo solicitaba directamente por su dirección daba click sobre algún vínculo.

La interactividad en este proceso era mínima debido a que el usuario sólo tenía el rol de receptor de la información y no tenía posibilidad de hacer peticiones personalizadas o dejar información propia.

Con el fin de mejorar tal situación se añadió más inteligencia tanto al cliente (navegador del usuario) como al servidor (emisor de páginas HTML).

En el lado del cliente se incorporaron los applets que son verdaderas clases de Java que se cargan y ejecutan en el navegador.

Del lado del servidor se incluyeron las páginas dinámicas de la mano de los formularios HTML. Se les llama dinámicas porque no existen en el disco duro del servidor, son generadas por éste haciendo llamadas a programas específicos conocidos como *scripts* que ejecutan instrucciones de recolección de datos de los formularios y acceso a bases de datos devolviendo como resultado al usuario la página construida para responder a su petición.

Los formularios HTML invierten el sentido del flujo de información ofreciendo al usuario una interfaz para enviar datos con la ayuda de controles simples como cajas de texto, botones de opción y selección.

El formulario transporta el nombre del programa en el servidor que será llamado para procesar los datos, recibirlos e introducirlos en la base de datos correspondiente y que a su vez enviará al cliente una página dinámica como respuesta indicándole que las tareas fueron realizadas con éxito, que se ha producido un error o que falta algún dato. La figura 1.12 muestra el esquema de funcionamiento de un script.

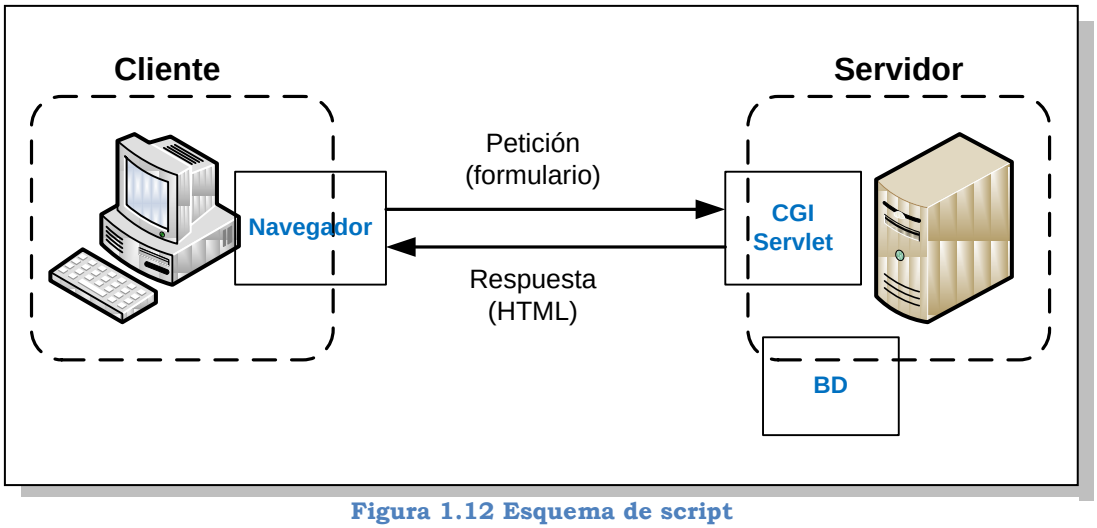


Figura 1.12 Esquema de script

El uso de scripts se debe a varias razones:

- Al residir en el servidor se logra la independencia del navegador.
- Tareas complejas se ejecutan más rápido del lado del servidor.
- Mayor seguridad en la ejecución bajo el control del administrador.
- La página Web se basa en los datos enviados por el usuario.
- La página Web contiene datos que cambian frecuentemente.

La primer tecnología usada en los scripts fueron los programas CGI (Common Gateway Interface) escritos en lenguajes como Perl, C y C++. A pesar de ser simples y ampliamente soportados presentaban la gran desventaja de que el servidor lanzaba una copia del programa por cada petición del usuario, por tanto si el sitio era muy visitado corría el riesgo de saturarse.

Java ofrece una tecnología alternativa que resuelve este problema con la programación de Servlets.

Los Servlets son programas en Java que se ejecutan en servidor Web o en el contenedor Web de un Servidor de Aplicaciones y fungen como una capa intermedia entre la petición de un cliente HTTP en un navegador Web y las bases de datos y aplicaciones del servidor devolviendo finalmente una respuesta al cliente.

Puede compararse con un applet que es cargado y ejecutado en el navegador, mientras que el Servlet es cargado y ejecutado por el servidor.

Ciclo de vida y finalidad del Servlet

1. Leer datos introducidos por el usuario en un formulario o applet.
2. Recoger petición en el servidor, si se observa un Servlet delega la acción al contenedor Web.
3. Obtener información adicional de la petición relacionada al cliente como navegador, cookies, nombre del equipo, etc.
4. Ejecutar el Servlet.
5. Generar resultados a partir de la comunicación con una base de datos, una llamada remota o invocando aplicaciones existentes.
6. Insertar la información resultante en una página HTML.
7. Establecer los parámetros adecuados del documento de respuesta.
8. Devolver el documento al cliente.

¿Qué se necesita para desarrollar Servlets?

Los Servlets son clases de Java que se codifican, compilan, instalan y ejecutan en un espacio restringido del servidor mediante un motor especial que se encargará de vigilar el ciclo de vida del Servlet, crearlo y destruirlo.

Por tanto se requiere instalar un motor de Servlets, en el caso de contar con el JDK en el equipo instalar el contenedor Web Jakarta-Tomcat que permite ejecutar Servlets y páginas JSP. En caso contrario, instalar el J2EE SDK que cuenta con el JDK y un servidor de aplicaciones.

Características:

- Extienden la portabilidad de Java al ser independientes del servidor y de su sistema operativo.
- Distribuyen el trabajo llamando a otros Servlets con tareas específicas redireccionando las peticiones.
- Fácil obtención de información del cliente y procesamiento de formularios.
- Uso de cookies y rastreo de sesiones.
- Actúan de enlace entre el cliente y una o más bases de datos.
- Permiten la generación dinámica de código HTML.
- No requieren soporte para Java en el navegador del cliente porque toda su ejecución se realiza del lado del servidor.
- Son cargados sólo la primera vez que se solicitan y permanecen en memoria. Las siguientes peticiones crean hilos de ejecución.

Las ventajas de los Servlets sobre CGI se observan en la tabla 1.1.

CGI	SERVLET
Si hay "n" peticiones simultáneas al mismo programa se carga "n" veces en memoria.	Una instancia del servlet en memoria y "n" subprocesos.
El programa finaliza cuando da respuesta a la petición lo que retarda los procesos y las conexiones a las bases de datos se pierden.	Permanece en memoria aún después de dar respuesta.
No existe comunicación directa con el servidor Web.	Optimización de conexiones a bases de datos y compartición de datos al comunicarse directamente con el servidor Web.
No hay verificación de código por lo que es propenso a ataques o accidentes.	Existe verificación de límites de matrices y protección de la memoria.

Tabla 1.1 Servlets vs. CGI

JSP

La tecnología de JSP (Java Server Pages) está orientada a la creación de páginas Web con la inclusión del lenguaje Java.

Como ocurre con la mayoría de las tecnologías de programación de páginas dinámicas, se utilizan 2 sintaxis diferentes para distinguir la lógica de la presentación de la lógica de la aplicación: la primera emplea etiquetas HTML que dan forma y aspecto final a la página Web representando el contenido estático; la segunda invoca el lenguaje de programación como VBScript en ASP o Java en JSP con etiquetas especiales que se mezclan en la página Web conformando la parte dinámica, como resultado de ambas partes se tiene un archivo con extensión *.jsp.

Cuando un cliente pide una página JSP del sitio Web por primera vez, la página es pasada al motor JSP y lleva a cabo su traducción en un Servlet que devuelve un archivo *.class. Posteriormente el motor de Servlets carga la clase del Servlet y lo ejecuta generando el HTML dinámico que es enviado al navegador del cliente. El proceso puede observarse en la figura 1.13.

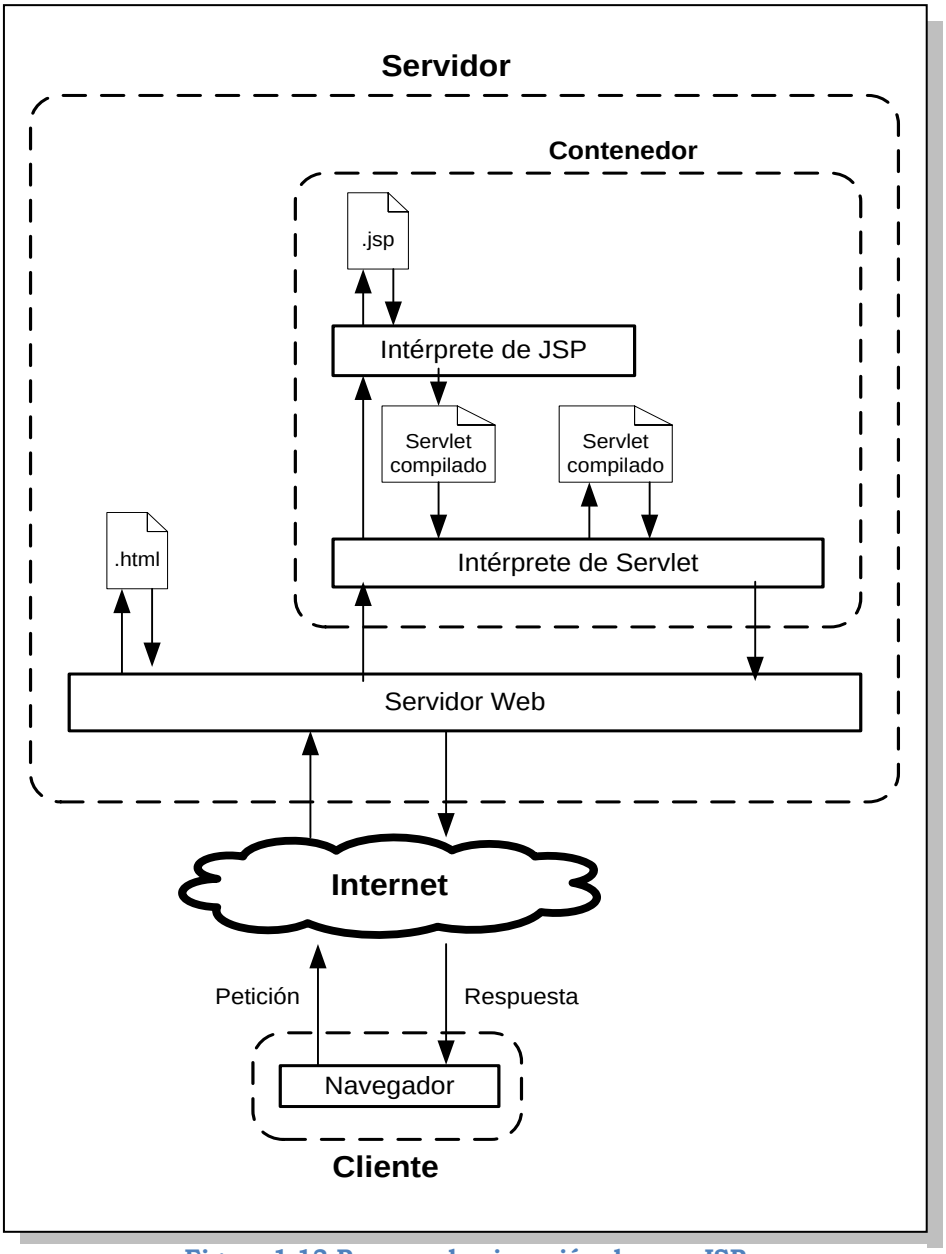


Figura 1.13 Proceso de ejecución de una JSP

Si la página es solicitada de nuevo el Servlet ya estará creado por lo que no es necesaria la conversión otra vez, a menos que el servidor detecte que la página de origen fue modificada, ésta volverá a ser compilada y convertida en Servlet.

En caso de detectar algún error, el contenedor envía un mensaje hacia el navegador del usuario.

Al emplear Java es imprescindible respetar las limitaciones y obligaciones del lenguaje a pesar de ser usado en el contexto de las páginas Web, no obstante cuenta con toda su potencia, portabilidad y agrega como ventaja la separación entre lo que se conoce como capa de presentación, capa de negocio y datos.

Al igual que los Servlets, la creación y ejecución de páginas JSP requiere de un entorno de desarrollo instalado y configurado correctamente, además de un servidor capaz de interpretar este tipo de páginas llamado contenedor que funciona sobre el servidor Web que recoge las peticiones y le pasa al contenedor las correspondientes a páginas JSP y Servlets.

Comparación con otras tecnologías

- *ASP (Active Server Pages).*

Páginas dinámicas escritas en VBScript (variante de Visual Basic) y JScript (variante de JavaScript) que carecen de portabilidad al estar atadas a sistemas operativos Windows y IIS. Actualmente forma parte de la plataforma .NET bajo la arquitectura cliente – servidor.

- *Servlets.*

Como se ha mencionado son programas en Java que son llamados para procesar datos y generar una respuesta pero su código es sólo visible y ejecutable del lado del servidor además de ser más complejo y laborioso, mientras que la página JSP devuelta al usuario mezcla el código Java con el HTML lo que podría considerarse inseguro pero más simple al separar el desarrollo en presentación y contenido.

- *PHP (PHP Hypertext Preprocessor).*

Lenguaje de tipo gratuito, opensource, multiplataforma y con soporte para las bases de datos más usadas que también se incrusta en las páginas Web dinámicas junto con etiquetas HTML.

- *JavaScript.*

A pesar de la similitud en el nombre no tiene relación con Java. Su dinamismo sólo se produce del lado del cliente y carece de soporte para programación de redes, formularios y bases de datos.

- *ColdFusion.*

Servidor de aplicaciones y lenguaje de programación para desarrollar páginas dinámicas, disponible para varias plataformas e interactúa con diversas bases de datos mediante SQL simple. No es de las más utilizadas pero es útil en la integración con otras aplicaciones como Flash para el que se creó originalmente.

JDBC

La gran parte de las aplicaciones y páginas Web hacen uso de las bases de datos desde que éstas surgieron para agendas, inventarios, validación de usuarios, publicaciones de periódicos en línea, información bancaria y catálogos de productos para comercio electrónico. Un simple sitio Web puede ofrecer gran cantidad de información y variedad de contenido gracias a una base de datos, algunas plantillas para páginas Web y el uso de páginas dinámicas.

Para llevar a cabo la conexión entre la base de datos y la interfaz de usuario en el sitio Web o en una aplicación se requiere un lenguaje de programación como Java que permita el enlace por medio de un driver JDBC.

JDBC (Java DataBase Connectivity) es un API de Java que permite la conexión con un manejador de bases de datos relacionales independientemente del tipo que éste sea; brinda al programador la facilidad de ejecutar instrucciones en el lenguaje estándar de consultas SQL (Structured Query Language) para crear, manipular y gestionar bases de datos.

Para emplear JDBC con un sistema manejador de bases de datos en específico se debe contar con el driver JDBC adecuado con el conjunto de clases distribuidas por el fabricante que haga la traducción de los mensajes de alto nivel de SQL a sentencias de bajo nivel propias de la base de datos usada. Con el API JDBC se deja atrás el escribir un programa que acceda a SyBase y otro para Oracle, con el driver correspondiente un único programa bastará para cualquier manejador soportado.

Existe otro estándar de conexión llamado ODBC (Open DataBase Connectivity) que pretendía conseguir en un inicio que el código fuera independiente del propietario de la base de datos pero al ser propiedad de Microsoft se limitaba a

ambientes Windows por lo que nunca logró plena portabilidad. Otras desventajas radican en que es el único medio de comunicación con algunos manejadores, es complejo de programar y está escrito en lenguaje C.

JDBC se basa en Java, en específico en los servlets para realizar 3 tareas:

1. Establecer la conexión a una base de datos ya sea o no remota.
2. Enviar sentencias SQL.
3. Procesar los resultados obtenidos de la base de datos.

Se emplea a los servlets por ubicarse en la capa media de una arquitectura de 3 capas, donde en todo momento se tiene el control de las operaciones con la base de datos, además porque corren del lado del servidor y no demandan la instalación de software en el cliente.

Tipos de drivers JDBC

La clasificación de drivers JDBC (figura 1.14) es elemental para conseguir un buen funcionamiento e interacción de la aplicación con la base de datos, se lleva a cabo a partir de dos preguntas: ¿el sistema de conexión está completamente escrito en Java? y ¿el cliente necesita un controlador adicional?

- *Tipo 1: Puente JDBC – ODBC.*

Se emplea cuando el manejador carece de soporte para JDBC y sólo dispone de drivers ODBC que si están disponibles en la mayoría de los sistemas. El puente realiza la conversión de los mensajes entre ambos pero es un proceso complejo, con demoras en el acceso, restricciones para red y presenta incompatibilidades de sus lenguajes de programación. Es preciso cargar el código binario del driver ODBC y el del manejador en la máquina cliente.

- *Tipo 2: API nativa.*

El driver contiene código Java que convierte las llamadas en métodos nativos de la base de datos, es decir, a la API propia de la base. En la máquina cliente debe cargarse el cliente de la base de datos como puede ser Sybase, Informix o DB2.

- *Tipo 3: Protocolo de Red.*

Las llamadas JDBC se traducen a un protocolo de red independiente del manejador y se comunican con una aplicación de capa intermedia en el servidor (middleware) que es capaz de conectar al cliente con diferentes manejadores de bases de datos. Es flexible, completamente desarrollado en Java y no necesita la instalación de software en el equipo cliente.

- *Tipo 4: Protocolo Nativo.*

Las llamadas se convierten en el protocolo de red usado por el SMBD. Es el mejor tipo debido a que está totalmente escrito en Java y comunica de forma directa la máquina cliente con el manejador. El driver es específico de la base que en la mayoría de los casos es proporcionado por el fabricante, por lo que si en algún momento debe migrarse de SMBD el driver también deberá cambiarse.

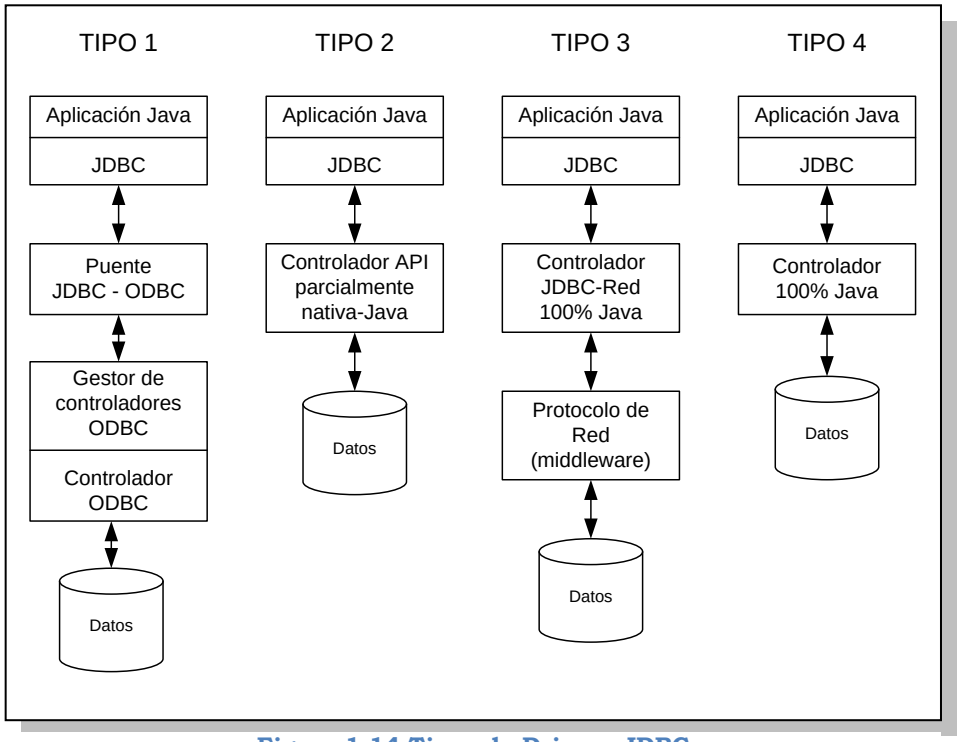


Figura 1.14 Tipos de Drivers JDBC

Ventajas de usar JDBC

- Las funciones y clases de la API son las mismas para cualquier manejador.
- El programador puede desarrollar una aplicación con un manejador sencillo y libre como MySQL, una vez probada llevarla a uno más potente como Oracle con la confianza de que el comportamiento será el mismo sin cambiar una línea de código.
- Actualización de múltiples registros con un sólo comando.
- Acceso a diversos manejadores en una misma transacción.
- Reutilización de conexiones a la base de datos en vez de una por comando.
- Soporta los modelos de acceso a bases de datos de 2 y 3 capas.
- Permite el paso de instrucciones propietarias del manejador aún cuando no respeten el estándar de SQL.
- Los resultados pueden ser devueltos como objetos y los posibles errores como excepciones.

1.3.3 Bases de datos

Una base de datos es una colección de datos clasificados y estructurados pertenecientes a un mismo contexto y almacenados sistemáticamente en uno o más archivos para su uso posterior.

Sistema manejador de bases de datos

Para almacenar y acceder a esta información existen los Sistemas Manejadores de Bases de Datos (SMBD o en inglés DBMS, DataBase Management System) que fungen como interfaz entre la base de datos, el usuario y las aplicaciones que la emplean. Son de utilidad en la manipulación de gran cantidad de información pero con las desventajas de un alto costo de hardware y de requerir varias personas para su administración.

Se distinguen 2 tipos de manejadores:

Libres:

- MySQL
- PostgreSQL

No Libres:

- Microsoft Access
- Microsoft SQL Server
- DB2
- Sybase ASE
- Oracle

Modelo de datos

Un modelo de datos es la descripción de los métodos para estructurar el almacenamiento y el acceso a la información.

Algunos modelos son:

- *Modelo Jerárquico.*

Las relaciones entre los datos son a partir de jerarquías en una estructura de árbol que inicia en un nodo raíz, donde cada nodo derivado o padre puede tener varios descendientes o elementos hijos sucesivamente hasta nodos sin hijos llamados hojas.

Se emplea cuando existe un gran volumen de información con datos compartidos que se relacionan entre sí mediante índices o punteros, o cuando los sistemas permanecen inalterables por mucho tiempo. Cabe señalar que es incapaz de manejar redundancia de datos.

- *Modelo en Red.*

Es una mejora del modelo jerárquico debido a que un hijo puede no tener padre o cualquier número de ellos.

Consta de dos conjuntos de datos: los registros y los enlaces que los relacionan.

- *Modelo Relacional.*

Utilizado en la actualidad para modelar situaciones reales y administrar datos dinámicos, permite la gestión de los datos en función de sus características lógicas sin tomar en cuenta la forma en que se almacenan a comparación con el modelo jerárquico y de red organizándolos en forma de tablas o relaciones.

La relación es un término que designa a una matriz de 2 dimensiones con filas y columnas. Cada columna de la tabla recibe el nombre de **atributo** o campo, mientras que las filas corresponden a los registros o **tuplas**.

Las tablas se enlazan entre sí mediante claves comunes, donde una clave es el conjunto mínimo de columnas que identifica a una fila dentro de una tabla.

La información contenida en la tabla puede manipularse a través de *consultas* escritas en un lenguaje estándar llamado SQL (Structured Query Language).

Normalización

Para el diseño de una base de datos relacional es imprescindible el proceso de normalización de los datos cuya finalidad es evitar la redundancia de datos y problemas de actualización, así como proteger la integridad de los datos.

Existen 5 formas de normalización de las cuales se explican 3 a continuación:

- *Primera forma normal (1FN)*

Un atributo o campo de una misma fila sólo puede contener un dato atómico e indivisible y no un conjunto de valores.

- *Segunda Forma Normal (2FN)*

Llamada de independencia funcional parte del cumplimiento de la primera forma y además marca que los atributos que no son clave deben depender completamente de la clave principal.

- *Tercera Forma Normal (3FN)*

Parte de la segunda forma normal y hace referencia a la existencia de una dependencia entre 2 atributos de una tabla donde uno será la clave de la tabla, visto de otra forma no debe existir dependencia funcional entre campos que no son clave.

Generalmente las bases de datos se normalizan hasta la tercera forma lo que representa un número alto de tablas con pocas columnas.

1.3.4 Sistemas operativos

Un sistema operativo son un conjunto de programas que actúan como intermediarios entre el usuario y el hardware con el fin de brindarle un ambiente para correr sus aplicaciones. Desempeña el papel de administrador de los recursos de hardware y software asignándolos de forma eficiente a programas y usuarios para la realización de sus tareas.

El sistema operativo (SO) combina la asignación de recursos con el control del hardware, que directamente no es fácil de usar y requiere una interfaz más específica como un software de aplicación.

Asimismo mantiene ocultos los procesos de administración mostrando al usuario una interfaz simple sin que le demande intervenir en cada tarea.

Sin embargo el SO no realiza por sí mismo todas las tareas, frecuentemente debe ceder el control al procesador orientándolo a que recursos usar y a que programas enfocarse temporalmente.

Las funciones del SO que son más utilizadas reciben el nombre de kernel o núcleo y se cargan en la memoria principal junto con las funciones empleadas en el momento, otros programas y datos del usuario.

De forma concreta un SO proporciona servicios para las siguientes áreas:

- Desarrollo de programas con utilidades del SO como herramientas de desarrollo y editores.
- Ejecución de programas.
- Acceso a dispositivos de entrada / salida.
- Acceso controlado a archivos mediante la identificación de dispositivos, estructura y permisos.
- Acceso al sistema con limitaciones por usuarios a recursos y datos específicos.
- Detección y respuesta a errores producidos por fallas de hardware o durante la ejecución de programas.
- Contabilidad y estadísticas para evaluar el rendimiento del equipo.

Los SO han evolucionado adaptándose a través del tiempo a los avances tecnológicos tanto en hardware donde todo se hace más compacto, simple y a un menor costo, como en software que es cada vez más potente robusto y demandante de recursos.

En el proceso de desarrollo de los SO surgieron algunos conceptos que marcaron las diferencias en sus versiones: administración de tareas, administración de usuarios y manejo de recursos.

Una tarea o proceso puede ser ejecutado únicamente de inicio a fin en un momento dado y hasta que finalice se ejecuta otro (monotarea), o varios que usan los recursos de forma alternada tan rápido que dan la apariencia de ocurrir al mismo tiempo (multitarea).

Un SO doméstico incluido en una PC sólo permite que un usuario a la vez inicie sesión y haga uso del sistema (monousuario), mientras que uno con orientado a servidores permite que varios usuarios ejecuten programas y empleen los recursos de forma simultánea pero garantizando la confidencialidad de la información y la actividad de cada uno (multiusuario).

Por último se dice que un SO es centralizado sí sólo utiliza los recursos del equipo sobre el que corre o distribuido si puede disponer de los dispositivos de otras computadoras conectadas en red.

El crecimiento de Internet ha impulsado el desarrollo de sistemas modernos que incluyen navegadores Web y aplicaciones para redes, comunicaciones y programación.

Dejando de lado los SO con fines domésticos de Windows y sus variantes para servidor, se observa que Sun Microsystems también ofrece soluciones de este tipo como Solaris.

Solaris es un sistema operativo de tipo UNIX que se basa SunOS y en System V Release 4 (SVR4) que funciona en arquitecturas SPARC, x86 (Intel y AMD), servidores y estaciones de trabajo.

Características:

- Procesamiento en tiempo real.
- Estructuras de datos dinámicamente asignadas.

- Gestión de memoria virtual.
- Sistema de archivos virtual.
- Núcleo multihilo.
- Soporte para multiprocesamiento simétrico (SMP) para arquitecturas con múltiples procesadores, éstos comparten los recursos y realizan las mismas funciones aumentando el rendimiento.
- Interfaz orientada a objetos para el sistema de archivos.

1.3.5 Arquitectura Cliente – Servidor

La arquitectura de red refiere al comportamiento que toman las computadoras conectadas a una red.

Originalmente la arquitectura predominante era la de un mainframe con una computadora central de gran tamaño y una serie de dispositivos de almacenamiento, periféricos y terminales conectados externamente.

Llevaba el nombre de **maestro/esclavo** porque las terminales o esclavos dependían totalmente del equipo maestro para realizar su trabajo con la desventaja de que cualquier fallo en éste repercutía en las terminales. Además el proceso para compartir información entre sistemas era muy complicado.

Posteriormente la arquitectura evolucionó dando paso a las redes de igual a igual y la arquitectura cliente servidor buscando resolver éstos problemas.

Redes de igual a igual

Todos los nodos o dispositivos tienen el mismo nivel de responsabilidad con la restricción de poseer también la misma capacidad de hardware.

Otras características son la comunicación bidireccional donde al ser nodos iguales las peticiones pueden hacerse por cualquiera.

Asimismo, un sistema puede prescindir de otro para funcionar con normalidad.

Arquitectura cliente servidor

Comparte un enfoque similar con la programación modular al considerar la división en módulos de un sistema para lograr rapidez, eficiencia y facilidad de mantenimiento.

Llevado a la arquitectura el módulo que genera las llamadas al sistema es el cliente mientras que el módulo que responde es el servidor, donde cada módulo corre sobre una plataforma de hardware diferente apropiada con la función que desempeña.

En este modelo se distinguen 2 procesos, uno para cada módulo que se comunican entre sí mediante protocolos estándar o particulares.

Cliente

El término cliente tiene a la vez dos acepciones, por un lado denota al nodo físico representado por el hardware, por otro hace referencia a la aplicación o software que corre sobre el nodo.

La aplicación cliente inicia diálogos con la aplicación servidor para solicitarle realice alguna tarea, suele contener la interfaz con el usuario, se encarga de validar datos ingresados y gestionar los recursos del hardware.

Servidor

Presenta 2 acepciones al igual que el cliente: nodo (hardware) y aplicación (software). La aplicación servidor ejecuta las tareas solicitadas por uno o varios clientes de forma simultánea, así como capturar y manipular la información de a base de datos.

Características de la arquitectura cliente servidor

- El cliente proporciona la interfaz y el servidor los medios para hacer uso de los recursos compartidos.
- Los procesos cliente y servidor tienen diferentes requerimientos de hardware.
- Demanda una conexión estable para funcionar bajo diferentes plataformas de hardware y software.
- Es escalable, tanto horizontalmente dado que acepta más nodos cliente, y en vertical con la posibilidad de cambiar a servidores más rápidos y potentes.

- Una aplicación cliente servidor se compone de 3 elementos:
  1. *Presentación.* Interfaz de usuario de tipo gráfica o un navegador Web.
  2. *Lógica de negocio.* Describe el comportamiento del sistema, los cálculos y procesos sobre los datos.
  3. *Gestión de datos.* Representada por el manejador de bases de datos que proporciona los medios para acceso y almacenamiento de información.

Se distinguen dos tipos de arquitectura cliente servidor: de dos y tres capas.

Modelo de 2 capas

El cliente se comunica directamente con el servidor por lo que es recomendable implementarse en ambientes con pocos clientes o de lo contrario puede provocar la saturación del servidor.

En el ámbito de las bases de datos se observa por ejemplo en un manejador de Microsoft Access que reside en el servidor mientras que una sencilla aplicación en Visual Basic en el cliente extrae datos de la base y se los presenta al usuario.

En caso de emplear JDBC, el driver específico para conectarse con el manejador debe instalarse del lado del cliente.

Visto de otra forma, los componentes de la arquitectura pueden agruparse de dos formas: la presentación en el cliente, la gestión de datos en el servidor y ambos compartir la lógica de negocio; la segunda la presentación y la lógica en el cliente y únicamente la persistencia de los datos en el servidor como se observa en la figura 1.15.

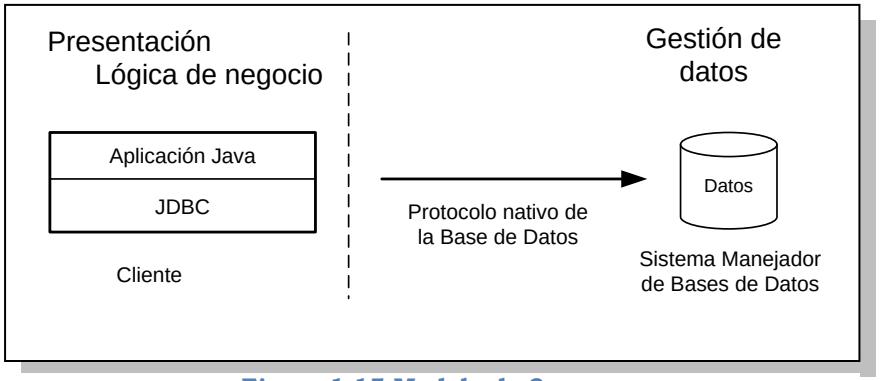


Figura 1.15 Modelo de 2 capas

Modelo de 3 capas

Introduce un elemento conocido como agente entre el cliente y el servidor cuyos objetivos son:

- Liberar de tareas al servidor.
- Traducir las peticiones de los clientes a lenguaje interno del servidor.
- Limitar el número de clientes que acceden.
- Redirigir peticiones a otros servidores.
- Recoger resultados.
- Enviar respuestas a los clientes.

La lógica de negocio toma mayor importancia y puede residir ahora en un servidor de aplicaciones, dejando la presentación en el cliente y la capa de gestión en un servidor de datos como se observa en la figura 1.16. Por ejemplo al usar drivers JDBC éstos ya no tienen que instalarse en la máquina cliente que únicamente contendrá la interfaz con el usuario. Las aplicaciones de tres capas presentan la ventaja de un mayor potencial de crecimiento y un sencillo mantenimiento.

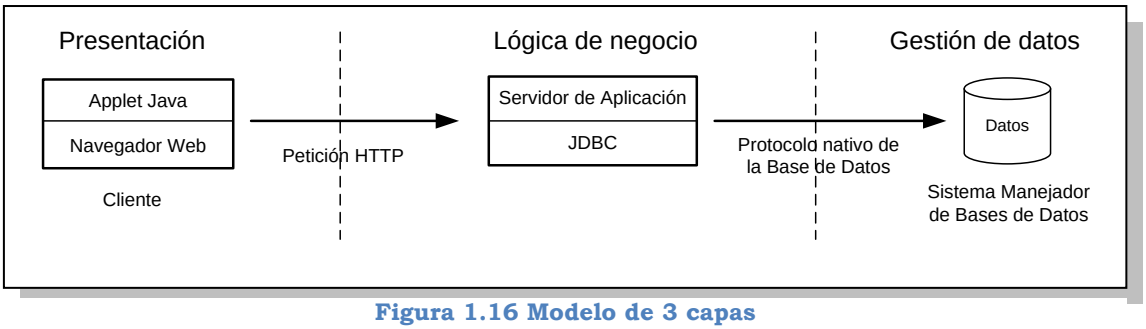


Figura 1.16 Modelo de 3 capas

Capítulo 2

Análisis del sistema

2.1 Propuesta de solución

A partir de la definición del problema y la descripción de la situación actual se plantea como solución el desarrollo de un sistema integral que reduzca las fallas y brinde un mejor servicio.

El sistema estará basado en una arquitectura cliente – servidor donde el elemento de presentación será el único del lado del cliente a través de una interfaz de usuario constituida por una página Web que sea visible por cualquier navegador Web evitando así la instalación de software adicional.

Las partes de lógica de negocio y gestión de datos se encontrarán del lado del servidor, donde un lenguaje de programación será la herramienta necesaria para crear los módulos encargados de la seguridad, de generar las páginas dinámicas que muestren y obtengan información, de darle persistencia en una base de datos, así como consultarla y modificarla, etc.

En concreto el sistema permitirá mediante un portal Web la gestión, asignación, visualización y revalidación de las calificaciones de laboratorios pertenecientes al Departamento de Ingeniería en Computación indicados como (L+) en el mapa curricular, lo que significa que deben inscribirse por separado en un horario diferente pero carentes de valor en créditos.

Las calificaciones serán asignadas cada semestre por los profesores de laboratorio (L+) a quienes se asignará un usuario y una contraseña para acceder vía Web al sistema, una vez confirmada su identidad podrán elegir los grupos que imparten con las listas de alumnos inscritos regularmente y ASDRI, así como las posibles calificaciones numéricas de 5 a 10 o NP que podrán indicar de forma sencilla a cada uno.

Para el caso de los ASDRI es indispensable realicen un proceso de registro por medio del cual serán incorporados a la lista de laboratorio y así puedan ser evaluados por el profesor como el resto de los alumnos. El trámite consistirá de forma similar al realizado en las teorías en descargar un formato disponible en el portal de asignación de calificaciones, llenar los campos requeridos con los datos personales del ASDRI y del laboratorio donde ha sido aceptado por el profesor quien deberá firmar el formato de registro para avalar la inscripción.

Una vez completado el formato con la información requerida éste será entregado en el Departamento de Ingeniería en Computación donde el administrador con acceso al sistema registrará al alumno en un periodo de tiempo coincidente con el proceso de alta de ASDRI en teoría. Cabe señalar que el profesor de laboratorio puede reservarse el derecho de aceptar o no alumnos en esta situación.

Una vez evaluados todos los grupos de laboratorio e incorporadas las calificaciones acumuladas de semestres previos como resultado de la revalidación, se procede a las opciones de visualización y descarga de calificaciones por los profesores de teoría que contarán también con privilegios de acceso al sistema mediante un usuario y una contraseña en el mismo portal Web.

El desarrollo del sistema y los elementos necesarios para su funcionamiento estarán a cargo del trabajo conjunto del Departamento de Ingeniería en Computación y la USECAD, donde el primero es responsable del proceso de calificaciones de laboratorio y su revalidación, mientras que la USECAD proveerá el alojamiento del sistema en uno de sus servidores y aportará las listas de alumnos de teorías y laboratorios resultantes del proceso de inscripción de cada semestre.

El software propuesto está ubicado dentro del contexto de un problema real por lo cual su análisis, desarrollo e implementación deben ser llevados bajo criterios formales dada su importancia, complejidad e impacto sobre las personas a las que está destinado el sistema.

Ingeniería de software

En vista de lo anterior el proceso del proyecto debe seguirse bajo el marco de la ingeniería de software concebida como el área de las ciencias de la computación que ofrece métodos, técnicas y herramientas para el desarrollo, operación y mantenimiento de software con el fin de crear un producto confiable y de calidad.

Se conforma por capas como se observa en la figura 2.1 donde la base es la calidad lograda bajo ciertos estándares y donde los requerimientos del cliente son satisfechos; la capa del proceso o también conocida como ciclo de vida del software define el marco de trabajo donde se aplican los métodos técnicos para obtener productos definiendo las actividades necesarias en las etapas de definición, desarrollo y mantenimiento.

El método especifica cómo construir técnicamente el software abarcando tareas como el análisis de requerimientos, el modelado del diseño que contempla las estructuras de datos, la arquitectura de programación y los algoritmos; la construcción del programa, las pruebas y el mantenimiento.

La capa de herramientas brinda el soporte automatizado o semiautomático para las capas de proceso y método logrando que la información procedente de una pueda ser usada en la otra.

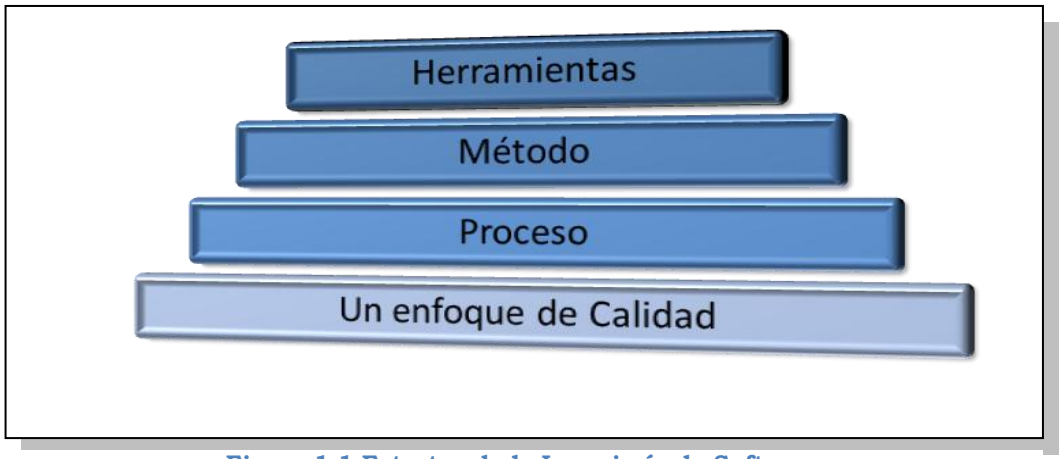


Figura 1.1 Estratos de la Ingeniería de Software

El proceso de desarrollo de software puede resumirse en 3 fases genéricas con diversas actividades cada una:

1. Definición: requisitos, análisis y planeación (¿qué?).
2. Desarrollo: diseño, codificación y pruebas (¿cómo?).
3. Mantenimiento: correcciones, adaptación y mejoras (cambio).

Además de abarcar las fases del ciclo de vida del desarrollo de cualquier sistema, la ingeniería de software considera aspectos de la gestión de proyectos como el manejo de personal, costos, etc.

Es recomendable en la creación de software incorporar una estrategia de desarrollo que defina el marco de actividades que acompañen a las fases genéricas, al proceso, los métodos y las herramientas, esta estrategia recibe el nombre de paradigma del software o modelo de proceso.

De acuerdo a las necesidades y naturaleza del proyecto se determinó emplear el modelo en espiral debido a su naturaleza de tipo evolutivo que toma en cuenta el cambio del software con el tiempo y los nuevos requerimientos del usuario, adaptándose y aplicándose a lo largo del ciclo de vida de la aplicación lo que resulta en versiones cada vez más completas.

El uso del modelo aporta además un proceso bien definido con mayor planeación, control y comunicación con el cliente.

El modelo espiral

El modelo en espiral propuesto por B. Boehm en 1988 combina la naturaleza iterativa de la construcción de prototipos (en donde tan pronto se completa una iteración o esfuerzo de desarrollo se procede a la siguiente iteración) con las características sistemáticas del modelo lineal secuencial pero añadiendo el análisis de riesgos.

El software se desarrolla en una serie de versiones incrementales a través de iteraciones, en cada una se atienden 4 aspectos:

- Determinar el objetivo a lograr.
- Plantear las alternativas.
- Analizar los riesgos y resultados esperados para cada una.
- Elegir la mejor alternativa evaluando sus alcances.

En la figura 2.2 se muestra la estructura de un modelo espiral típico que se compone del eje del punto de entrada en cuyo centro parte la espiral que gira en el sentido de las manecillas del reloj donde el primer circuito puede producir la especificación del producto que es usada en giros posteriores para el desarrollo del prototipo y progresivamente mejores versiones.

El modelo se divide en un conjunto de actividades llamadas regiones de tareas que pueden ir de 3 a 6 que se adaptan a las características del proyecto y son:

1. Comunicación entre el equipo de desarrollo y el cliente.
2. Planificación de recursos y tiempos.
3. Análisis de riesgos técnicos y de gestión.
4. Ingeniería o construcción de modelos de la aplicación.
5. Construcción, pruebas, instalación, entrega de documentación y soporte al usuario.
6. Evaluación y reacciones del cliente ante cada versión entregada resultado de la ingeniería y las pruebas.

Figura 2.2 Modelo espiral típico

Cada ciclo de la espiral comienza con la recolección de requerimientos y la identificación de objetivos, luego se realiza un análisis de riesgo evaluando las alternativas respecto a los objetivos y restricciones; posteriormente se determina si es conveniente seguir a pesar de los riesgos o se plantean estrategias para resolverlos; se obtiene un prototipo que es presentado al cliente y una vez resueltos los riesgos se sigue el ciclo en cascada.

El giro concluye con la revisión del ciclo anterior y la planeación del siguiente hasta que la evaluación del cliente sea positiva al cumplirse todos los requerimientos.

Ventajas:

- Enfoque más realista de ingeniería de software.
- Permite reducir riesgos en cada ciclo.
- No requiere una definición completa de los requisitos para comenzar a funcionar.
- Incorpora planeación, objetivos de calidad y gestión de riesgos.
- Permite vuelta atrás al identificar problemas.

Desventajas:

- Requiere de experiencia en la identificación de riesgos.
- Necesita participación activa del cliente.
- Si un riesgo importante no se descubre y atiende causará problemas.
- Es difícil convencer al cliente que es un modelo estable.

2.2 Beneficios de la automatización

Una vez desarrollado, probado e implementado el sistema propuesto se logrará un impacto benéfico en profesores de teoría y laboratorios del tipo L+ del Departamento de Ingeniería en Computación, alumnos que cursen estas asignaturas como regulares o ASDRI, y en el personal del Departamento responsable del proceso de calificaciones y revalidaciones de laboratorios. Las ventajas estimadas del sistema son:

- Los profesores de teorías y laboratorios tendrán acceso a una interfaz sencilla para asignar o visualizar calificaciones según sea el caso.
- Con el registro de los ASDRI se tendrá un mejor control de su inscripción en los grupos de laboratorio.
- Los datos de calificaciones serán conservados en la base de datos acumulándose con los de semestres anteriores para cumplir con la revalidación.
- Reducción de errores humanos por parte de los profesores y del personal del Departamento.
- El proceso de asignación, registro, visualización y revalidación será automático, confiable y seguro.
- El profesor de laboratorio ya no escribirá los datos de sus alumnos y sus calificaciones en un archivo para después enviarlo por correo reduciendo el factor de error.
- Ya no será necesaria la verificación del archivo de calificaciones de cada profesor para completar datos, estandarizar o redondear calificaciones.
- El sistema mostrará al profesor de laboratorio su información precargada, la lista de alumnos regulares y ASDRI con sus números de cuenta y las calificaciones válidas sólo en espera de seleccionar la que corresponda a cada uno.
- Se eliminan los errores relacionados con la escritura de la dirección de correo electrónico o del servidor encargado de gestionarlo derivados del envío del archivo de calificaciones al ya no ser éste necesario.
- Contemplará servicios de seguridad como la confidencialidad de la información que viaje por la red y la verificación de identidad de los usuarios que ingresen al sistema.
- El sistema será abierto a los profesores de laboratorio para calificar durante un lapso de tiempo definido y comunicado a todos a principio del

semestre con el fin de garantizar que el proceso esté completo en las fechas establecidas.

- Los profesores de teoría tendrán a su disposición oportunamente las calificaciones de laboratorio de su grupo para realizar su evaluación final sin demoras como sucede en el método actual.
- Consulta y corrección de información de forma rápida y sencilla bajo reglas de control de acceso.
- Se llevará un registro de las correcciones de calificación gracias a la entrega al Departamento de Computación del formato correspondiente descargable del portal Web.

2.3 Requerimientos

Los requerimientos identifican las necesidades de funcionalidad y comportamiento del producto, restricciones de diseño e interfaces de usuario, además ayudan a comprender el impacto del software y lo que realmente quiere el cliente.

La etapa de análisis permite a través de la comunicación directa con el cliente definir las funciones del software, el flujo de los datos del sistema al usuario o sistemas externos, su interacción y su almacenamiento.

Esta etapa es indispensable en el planteamiento y resolución efectiva de problemas al fijar la base del diseño a partir del entendimiento de lo que el cliente necesita, obteniendo así una descripción de los requisitos y lista de funcionalidades en lenguaje natural, especificaciones o escenarios de usuario que posteriormente podrán ser verificados durante el proceso en reuniones con el cliente.

2.3.1 Sistema

Categorías de usuarios del sistema

- Profesor Teoría (PT). Accede al sistema para visualizar y descargar las listas de los grupos que imparta con las calificaciones de laboratorios incluidas.
- Profesor de Laboratorio (PL).
- Alumno sin derecho a reinscripción (ASDRI).
- Administrador de USECAD (AU).
- Administrador del Departamento de Computación (ADC).

Requerimientos generales

- Crear un portal que permita asignar, revalidar y visualizar calificaciones de laboratorios L+ para alumnos inscritos regularmente y ASDRI (serán registrados por el sistema) del Departamento de Ingeniería en Computación las cuales serán almacenadas para su posterior consulta, descarga y corrección. Ciertas funciones estarán disponibles por periodos de tiempo determinados por el Departamento de Computación.

- Únicamente las teorías con laboratorios L+ coordinados por el Departamento de Computación serán contemplados en el funcionamiento del sistema y son:

  Plan de Estudios 2005
  - ✓ Computación para Ingenieros.
  - ✓ Programación Avanzada y Métodos Numéricos.
  - ✓ Administración de Redes.
  - ✓ Computación Gráfica.
  - ✓ Diseño de Sistemas Digitales.
  - ✓ Dispositivos de Almacenamiento Entrada / Salida.
  - ✓ Redes de Datos.

  Plan de Estudios 1994
  - ✓ Microcomputadoras.
  - ✓ Memorias y Periféricos.

- El acceso al sistema por parte de los usuarios PT, PL y ASDRI se efectuará desde cualquier navegador Web sin importar el sistema operativo del equipo de cómputo mientras cumpla con las condiciones mínimas indicadas más adelante, ya sea desde su casa, un local de renta de computadoras o salas de cómputo de la Facultad.
- Las interfaces para asignación de calificaciones deben ser semejantes a las mostradas por el sistema existente para asignaturas teóricas con la finalidad de hacerlas más amigables y aceptadas por los usuarios PL.
- El diseño de las interfaces debe estar orientado a que los usuarios digiten la menor información posible para reducir errores.
- Dada la naturaleza del proyecto no se cuenta con presupuesto para la compra de equipo o el pago de licencias de software, por lo tanto el desarrollo e implementación deberá cubrirse con aplicaciones de distribución libre.
- La administración del sistema estará a cargo de personal de USECAD y del Departamento de Ingeniería en Computación.
- El proyecto deberá operar sin problema alguno en el servidor de USECAD garantizando la compatibilidad con su arquitectura, sistema operativo y sistema manejador de bases de datos.

Requerimientos funcionales por módulos

- *Registro de ASDRI.*

Módulo dedicado al alta de ASDRI en las listas de laboratorios a partir de la información contenida en el formato descargable y entregado al Departamento de Computación por el ASDRI. Los datos son:

- Nombre del Alumno.
- Número de Cuenta.
- Clave del Laboratorio.
- Nombre del Laboratorio.
- Grupo del Laboratorio.
- Nombre del Profesor de Laboratorio.
- Firma del Profesor de Laboratorio.
- Grupo de Teoría.

- *Acceso al sistema.*

Módulo destinado a validar el usuario y contraseña de los usuarios PT, PL AU y ADC introducidos en la interfaz Web.

- *Alta de usuarios.*

Módulo en el cual los administradores podrán crear usuarios PT y PL ante la incorporación de nuevos profesores cada semestre para impartir alguna de las asignaturas abarcadas por el sistema.

- *Descarga de formatos.*

Módulo destinado a proporcionar los formatos en el portal sin necesidad de iniciar sesión en el sistema y podrán ser descargados para la corrección de calificaciones de laboratorio o el registro de ASDRI.

- *Consulta de listas de PL.*

Módulo que muestra en una interfaz Web por grupo la lista de alumnos inscritos y ASDRI registrados.

- Consulta de listas de PT.

Módulo que muestra en una interfaz Web por grupo la lista de alumnos inscritos y ASDRI con su calificación de laboratorio.

- *Corrección de calificación.*

Módulo dedicado a la modificación de calificaciones de laboratorio posterior al proceso de asignación llevando una bitácora de cada petición realizada mediante el formato descargable del portal y entregado al Departamento de Computación por el usuario PL con los siguientes datos:

- Nombre del alumno.
- Número de cuenta.
- Clave del laboratorio.
- Nombre del laboratorio.
- Grupo de laboratorio.
- Nombre del profesor de laboratorio.
- Firma del profesor de laboratorio.
- Exposición de motivos del profesor de laboratorio para la corrección.
- Calificación asignada erróneamente.
- Calificación correcta.

- Calificación.

Módulo encargado de mostrar en una interfaz Web por grupo de laboratorio la lista de inscritos con ASDRI y los campos con las calificaciones válidas frente a cada alumno para seleccionarla según corresponda sin necesidad de que el usuario PL la ingrese por teclado. Al concluir la información será almacenada en la Base de Datos.

- *Distribución grupal.*

Módulo de búsqueda de calificación de alumno asignada en el presente semestre u obtenida en semestres previos para ser revalidada, en caso de existir más de una se elegirá bajo el siguiente criterio:

Se asignará la calificación aprobatoria mayor sin importar el semestre de procedencia, de no existir tomar la más reciente sea o no aprobatoria.

El proceso se repite sucesivamente para todos los alumnos de un grupo de teoría y así para todos los grupos y asignaturas reconocidas por el sistema.

El módulo puede ejecutarse automáticamente y por petición del usuario ante la corrección de calificaciones.

- *Descarga de listas.*

Módulo para crear y proporcionar la descarga de archivos de hoja de cálculo con los datos del profesor y la asignatura, pero además la lista de teoría por grupo con calificaciones de laboratorio, o la lista de laboratorio por grupo completa con inscritos y ASDRI.

- *Interfaz usuario PT.*

Módulo encargado de generar la interfaz Web propia del usuario PT mostrando sus datos, así como las opciones de consulta y descarga de listas por grupo de teoría que incluyen las calificaciones de laboratorio.

- *Interfaz usuario PL.*

Módulo encargado de generar la interfaz Web propia del usuario PL mostrando sus datos, así como las opciones de consulta y descarga de listas por grupo posterior al registro de ASDRI, pero además la opción de calificar listando los grupos impartidos en el semestre e indicando gráficamente el estado pendiente o exitoso del proceso para cada uno.

- *Interfaz usuario ADC.*

Módulo enfocado al usuario de tipo administrador para el cual debe generarse una interfaz Web diferente al resto de los usuarios, debe incluir mayores parámetros de consulta y opciones que sólo éste tipo de usuario puede realizar como el registro de ASDRI, correcciones de calificación, creación de archivos con calificaciones finales, respaldos, alta de usuarios, etc.

- *Consulta de usuario ADC.*

Módulo de consultas de calificación de laboratorio por parte del administrador bajo los siguientes criterios de búsqueda:

- *Por grupo de teoría.* Mostrando todos los alumnos, su calificación y el grupo de laboratorio del que fue tomada.

- *Por alumno.* Ya sea por su nombre o número de cuenta y filtrando además por semestre y asignatura.
- *Por grupo de laboratorio*. Mostrará la lista disponible a visualizar y descargar por el profesor de laboratorio.
- *Por semestre*. Mostrando todos los resultados o filtrados por asignatura.

- *Respaldo periódico de información.*

Con el objetivo de evitar la pérdida de información por fallas del sistema u ocurridas en el servidor relacionadas con el sistema operativo, con el sistema manejador de bases de datos o con el disco duro, se requiere de un módulo que realice respaldos de información vital como el acumulado de calificaciones en otras tablas dentro del gestor o en medios externos a petición del usuario y automáticamente por periodos de tiempo establecidos.

- *Respaldo del sistema.*

Módulo responsable de la creación del respaldo del propio sistema en caso de sufrir alteraciones y sea necesario restablecer versiones previas más estables o ante mejoras posteriores.

2.3.2 Hardware

Servidor SunFire v880

El proyecto está destinado a operar en un ambiente Web donde los usuarios accedan al sistema a través de cualquier equipo conectado a Internet, por tanto el sistema debe estar alojado en un servidor de alto desempeño, seguro y confiable que integre además un gestor de bases de datos.

Tal servidor no será adquirido sin embargo la USECAD ha puesto a disposición uno de sus servidores que cumple con todas las condiciones y que además contiene la información necesaria para el completo funcionamiento del sistema.

El servidor en cuestión es un Sun Fire modelo v880 de alto desempeño, memoria compartida, escalable, con sistema de multiprocesamiento simétrico y soporte de hasta 8 procesadores Sun UltraSparc III, haciéndolo ideal para estaciones de trabajo y hospedaje de sitios Web.

Su poder de procesamiento es proporcionado por un máximo de 4 tarjetas CPU/Memoria, cada tarjeta incorpora:

- 2 procesadores UltraSparc III.
- 8 Megabytes de memoria caché por cada procesador.
- 16 ranuras para módulos de memoria con capacidad máxima por cada una de 1GB.
- A lo anterior se añaden otras características de almacenamiento y conectividad:
- Máximo de 12 discos duros de 18, 36, 72 o 146 GB.
- Canal de fibra óptica y SCSI para almacenamiento.
- Protocolos de red Ethernet, Fast Ethernet y Gigabit Ethernet.
- Unidad óptica DVD-ROM.
- 9 ranuras de expansión PCI, 7 x 33 MHz y 2 x 33 o 66 MHz.
- Interfaces USB y serial.
- RSC o Remote System Control, es una tarjeta de administración segura de servidor sobre líneas de modem o red para monitorear sus condiciones de ambiente, temperatura y alimentación, controlar funciones mediante consola del sistema como diagnósticos, encendido y reinicio, emitir notificaciones de problemas.

El procesador UltraSparc III implementa el SPARC V9 Instruction Set Architecture (ISA) y el Visual Instruction Set (VIS) para acelerar los procesos multimedia, de redes, encriptación y procesamiento de Java. Desarrollado por Sun MicroSystems y fabricado por Texas Instruments basado en el SPARC, ésta fue la primera arquitectura RISC abierta con especificaciones de diseño disponibles y presentaba un número reducido de instrucciones. Algunas otras características del UltraSparc III son:

- Microprocesador super-escalar.
- Rango de operación de 750 a 1200 MHz.
- Controlador de memoria integrado y bus de multiproceso dedicado para lograr el rendimiento de multiprocesamiento de memoria compartida.
- Interfaz externa con un bus de datos de 128 bits y bus de direcciones de 43 bits operando a 150 MHz.
- unidades aritmético lógicas (ALUs), 2 unidades de punto flotante y 1 unidad de carga y almacenamiento (LOAD/STORE).
- Transferencia de hasta 9 GB de información por segundo.

- Procesamiento de 64 bits.
- Detección y corrección de errores en cachés.
- 29 millones de transistores.

Equipo cliente

Los usuarios requieren una PC con acceso a Internet cuyas especificaciones satisfagan la carga de procesos de navegadores Web para acceder al sitio pero se recomiendan los siguientes requisitos para no afectar el desempeño:

- *Disco duro:* no hay requisitos debido a que el sistema no se instala en los equipos cliente.
- *Memoria RAM:* 256 MB (mínimo) y 512 (recomendado) para Microsoft Windows XP, 1 GB (mínimo) y 2GB (recomendado) para Microsoft Windows Vista y 7.
- *Procesador:* 750 MHz (mínimo) equivalente a Pentium III en adelante.

Equipo de desarrollo

La instalación de herramientas y entornos de desarrollo, de sistemas manejadores de bases de datos, así como el software que emule el comportamiento de un servidor Web demandan los siguientes recursos de cómputo:

- *Disco duro:* 80 GB (mínimo) para las instalaciones y bases de datos de prueba.
- *Memoria RAM:* 1 GB (mínimo) o 2 GB (recomendado).
- *Procesador:* Pentium 4 (mínimo) o doble núcleo (recomendado).
- *Red:* Tarjeta de red 10/100 Mbps.

2.3.3 Software

El desarrollo del sistema para la gestión de calificaciones de laboratorios y su revalidación tiene una naturaleza Web, que involucra lenguajes de programación, creación de interfaces de usuario, así como el almacenamiento y consulta de información, por tanto son indispensables herramientas de software que cubran los aspectos del sistema en todas sus etapas, desde entornos de desarrollo hasta servidores Web y manejadores de bases de datos.

Existen diversas herramientas de tipo privativo o libre especializadas en cada área descritas a continuación.

Java

Java no es sólo un lenguaje de programación, constituye también las herramientas para desarrollarlo y ejecutarlo, es decir, incluye el entorno de programación (API y compilador) y el entorno de ejecución (JRE).

El entorno de desarrollo necesario es el JDK de Sun o Java Development Kit y en la versión 2 toma el nombre de J2SDK.

La plataforma Java 2 se divide en 3 categorías dependiendo el mercado al cual está enfocado:

- *Java 2 Standard Edition (J2SE)*

Orientado a equipos de escritorio para el desarrollo de aplicaciones y applets que poseen generalmente una interfaz de usuario.

- *Java 2 Enterprise Edition (J2EE)*

Toma al J2SE y añade clases para el desarrollo de aplicaciones empresariales orientadas a servidores basadas en Enterprise JavaBeans, Servlets, Java Server Pages y Extensible Markup Language (XML).

- *Java 2 Micro Edition (J2ME)*

Enfocado a dispositivos electrónicos con pocos recursos como teléfonos celulares, PDA's, sistemas GPS, etc., que poseen poca memoria y baja capacidad de procesamiento, así como limitadas resoluciones gráficas de pantalla.

IDE (Integrated Development Environment)

El IDE o Entorno de Desarrollo Integrado es una aplicación dedicada a la creación de proyectos de software bajo uno o más lenguajes de programación y consta de una serie de herramientas como un editor de texto de código fuente, un compilador, un depurador y un constructor de interfaces gráficas de usuario GUI brindando al desarrollador un marco de trabajo completo y más amigable.

Eclipse

Es un IDE de código abierto basado en plugins creado originalmente por IBM para el desarrollo de aplicaciones Java, es ideal para la gestión de proyectos grandes con un buen número de clases y cuenta con una versátil opción de depuración de programas.

Está disponible en la página de Fundación Eclipse en diferentes versiones compiladas que agrupan herramientas útiles para situaciones o lenguajes de programación específicos.

Su instalación es bastante sencilla, la versión seleccionada se descarga en un archivo comprimido con extensión zip, éste se descomprime en la carpeta deseada y sólo es necesario invocar el archivo ejecutable eclipse.exe para iniciar la aplicación, cabe señalar que es indispensable tener instalado previamente el J2SDK.

Algunas otras características de Eclipse son:

- Multiplataforma (Linux, Windows, MacOS).
- Soporte de Arquitecturas x86 y 64.
- Estructura de plugins para agregar nuevas características o compatibilidad con otros lenguajes de programación como C, C++ y PHP.
- Control de versiones con CVS o con subversion.
- Resaltado de sintaxis y autocompletado.
- Compilación en tiempo real.
- Asistentes o Wizards para la creación de proyectos, clases y platillas de código.

NetBeans

Es un IDE de carácter libre para escribir, compilar, depurar y ejecutar programas en Java. Tiene una base modular que le permite extenderse para cubrir otro tipo de aplicaciones o desarrollar herramientas propias.

Fue fundado por Sun MicroSystems y en la actualidad cuenta con una gran comunidad de usuarios que aportan módulos que interactúan con las APIs de NetBeans y pueden ser incorporados para extender el IDE.

Su descarga inicial ya contiene los módulos con las funciones orientadas al soporte de Java y para el sistema de control de versiones. Presenta algunas otras características importantes:

- Multiplataforma (Windows, MacOS, Linux, Solaris).
- Amplia fuente de módulos de terceros.
- Soporte para lenguajes C, C++, PHP, JavaScript, Ajax.
- Gratuito y sin restricciones de uso.
- Herramientas de esquemas XML.
- Soporte a persistencia.
- Desarrollo de aplicaciones empresariales.

Dreamweaver

Aunque no es considerado un IDE como tal, es una aplicación útil para la construcción bajo estándares de páginas, sitios y aplicaciones Web, fue creado por Macromedia y está cargo actualmente de Adobe Systems.

Además del editor de código añade un entorno gráfico más amigable a los diseñadores para crear visualmente páginas Web estáticas y dinámicas mediante la inserción de elementos dispuestos en paneles de herramientas, tal peculiaridad ubica también la aplicación en el ámbito de los editores "lo que ves es lo que obtienes" (WYSIWYG).

Algunas otras características importantes son:

- Integración con Adobe Flash y Adobe Fireworks.
- Soporte de tecnologías CSS y JavaScript.
- Disponible para las plataformas MacOS y Windows.
- Conexión a bases de datos MySQL y Microsoft Access.
- Contenido dinámico con ASP, ASP.NET, ColdFusion, JSP y PHP.
- Interfaz personalizable.
- Cliente de FTP integrado.
- Comparación de contenido en diferentes navegadores.
- Integración de contenido interactivo y videos de forma sencilla como FLV.

Servidores Web

Un servidor Web hace referencia al programa y al equipo donde se ejecuta encargado de transferir páginas Web (generalmente en código HTML) con textos, vínculos y contenido incrustado como imágenes, animaciones, audio y video.

El servidor se mantiene a la espera de peticiones a los sitios que proporciona, el usuario solicita la página indicando su dirección en la red o URL (Uniform Resource Locator) en un navegador Web entablando comunicación por el protocolo HTTP, si el recurso se encuentra disponible el servidor responde con el código HTML de la página que es recibido e interpretado por el navegador y la muestra finalmente en pantalla.

Sin embargo, un servidor Web no se limita a la distribución de páginas HTML, dado que un navegador puede invocar una aplicación integrada por fragmentos de código que se ejecuta a partir de la petición o como parte de la respuesta tanto del lado del cliente o del lado del servidor.

En tal situación el servidor recibe el nombre de contenedor Web o contenedor Servlet/JSP, que puede operar por sí mismo respondiendo con recursos estáticos, dinámicos y aplicaciones, o bien mediante un conector trabajar en conjunto con un servidor HTTP encargado de responder o re-direccionar al contenedor según sea el caso.

Servidor HTTP Apache

Servidor Web de código abierto desarrollado bajo la Apache Software Foundation que ha sido empleado por gran parte de los sitios Web del mundo debido a su facilidad de instalación y a la posibilidad de convertir cualquier PC de bajos recursos en servidor corriendo bajo sistemas Unix o Windows.

Es un software de arquitectura modular organizado en módulos base que contienen las funciones elementales y los de multiproceso responsables de los puertos y las peticiones.

Si se desean añadir utilidades basta con agregar los módulos necesarios sin volver a instalar el software que en la mayoría de los casos corre sin realizarle cambios, sin embargo es altamente configurable a pesar de la carencia de una interfaz gráfica apropiada para ello. Otras características son la capacidad de conectarse directamente con una base de datos y la inclusión de módulos

compatibles con los lenguajes PHP, Perl y Python para el desarrollo de aplicaciones o funciones avanzadas.

Su versión 2 ofrece mejoras respecto a versiones previas:

- Un nuevo conjunto de interfaces de programación de aplicaciones (API's).
- Los módulos pueden actuar como filtros de contenido.
- Soporte para el formato de direcciones IPv6.
- Las páginas de respuesta a errores ahora se presentan en diversos idiomas.
- Es más rápido y estable en sistemas que no son tipo Unix debido a la incorporación de los módulos de multiprocesamiento y API nativa para cada plataforma.
- Posibilidad de servir en distintos protocolos.

Existen paquetes para el desarrollo de aplicaciones como WampServer y EasyPHP de gran ayuda para la fase de pruebas antes de su implementación final que integran Apache como servidor Web junto con el manejador de bases de datos MySQL y el lenguaje de programación PHP en una misma interfaz, son de fácil instalación y configuración para sistemas Windows.

Apache Tomcat

También llamado Jakarta Tomcat es un servidor Web de código abierto diseñado para trabajar con Java, desarrollado originalmente por Sun como parte del Java Server Web Development Kit y a cargo actualmente de la Apache Software Foundation.

Es un contenedor de Servlets que implementa las especificaciones de Sun MicroSystems, no es más que un Shell de ejecución que maneja e invoca Servlets por cuenta del usuario o los convierte resultado de la compilación de JSP.

Puede funcionar como servidor Web independiente, sin embargo no es tan rápido para servir páginas estáticas como Apache, no es tan configurable ni maneja otros lenguajes además de Java, por lo cual su uso autónomo se recomienda para entornos de desarrollo o para sitios bajo condiciones mínimas de transacciones y velocidad.

Dado lo anterior se recomienda emplear en forma de plugin para otros servidores Web ya sea Apache o IIS. De esta forma se complementa la

funcionalidad del servidor Web con las ventajas que implica el contenedor como threading, manejo de sesiones, conectividad, etc.

Ambos comportamientos de Tomcat se observan en la figura 2.3:

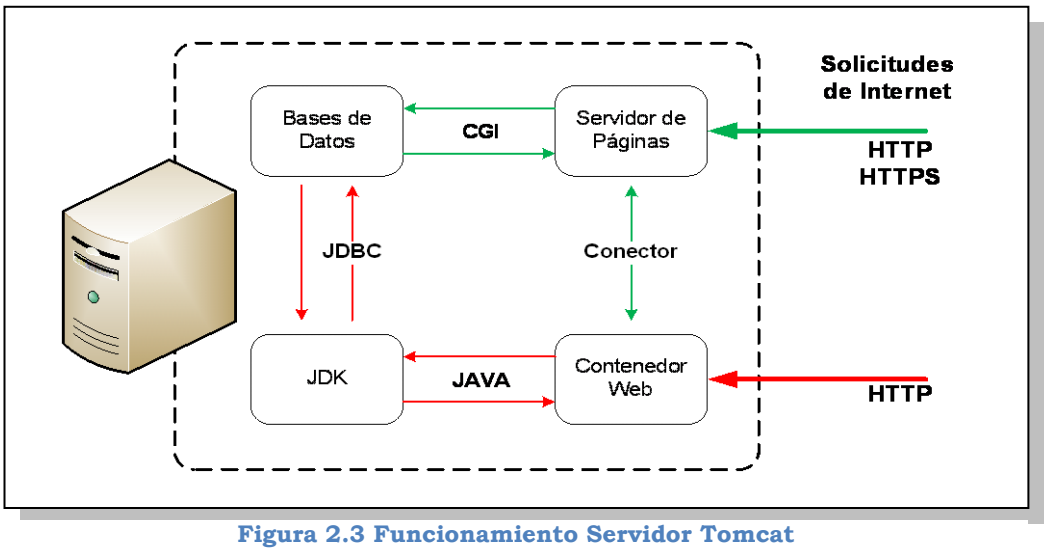


Figura 2.3 Funcionamiento Servidor Tomcat

Sistema Manejador de Bases de Datos (SMBD)

El SMBD es el software que sirve de interfaz entre la base de datos física, el usuario y las aplicaciones que la utilizan. Permite crear, definir y estructurar una base de datos, guardando la información en algún medio controlado por el SMBD para después consultarla o modificarla.

Existen productos libres o con licencia propietario disponibles que cubren su función primordial pero además incorporan mecanismos de seguridad, respaldo y recuperación, manejo de transacciones y tiempo de respuesta variable.

Sybase ASE

Sybase es una compañía que crea productos y servicios dedicados a la gestión de información, herramientas de desarrollo y almacenamiento de datos.

En el ramo de los SMBS relacionales cuenta con Adaptive Server Enterprise (ASE), reconocido como un sistema altamente escalable de alto rendimiento y

bajo costo, con soporte a grandes volúmenes de datos, transacciones y usuarios. Está disponible en las siguientes ediciones:

- ASE Enterprise Edition
- ASE Small Business Edition
- ASE Developer's Edition
- ASE Express Edition para Linux
- ASE Cluster Edition

En su versión empresarial 15 se distinguen opciones de seguridad únicas y funciones que incrementan el rendimiento reduciendo los costos y riesgos de operación mientras responde a las exigencias de tamaño de varios terabytes y millones de transacciones por minuto en diversas plataformas. Como parte de estas funciones están:

- In-Memory. Es una opción enfocada a la virtualización de los datos y el escalamiento crítico de grandes volúmenes de datos y usuarios concurrentes para ambientes que demandan tiempos de respuesta inmediatos.
- Servicios de recuperación más rápidos con menor uso de recursos de red y almacenamiento.
- Posibilidad de 4 tipos de particionamiento de datos ubicados en varios dispositivos físicos, si uno de éstos falla las particiones restantes pueden usarse.
- Sistema de encriptación a través de permisos de usuarios.
- Conexiones seguras SSL.
- Soporte a herramientas de desarrollo y lenguajes como Power Builder, Visual Basic, Java, PHP, etc.
- Auto-administración, manejo automático de recursos y bases de datos transportables.
- Soporte a protocolos ODBC, ADO.NET, JDBC y OpenClient de Sybase.
- Mejoras en servicios de Java y XML.
- Tecnología de consultas para transacciones más inteligentes.
- Cómputo distribuido con tecnologías grid y cluster.
- Arquitectura orientada a servicios Web.
- Compatibilidad inalámbrica.

2.4 Criterios de elección de software

Las herramientas de software fueron elegidas considerando en primer lugar los requisitos restrictivos del servidor donde será colocado el proyecto y el sistema manejador de bases de datos. El resto de las herramientas se seleccionaron de acuerdo a la compatibilidad entre ellas, desempeño y seguridad.

A continuación se muestran en la tabla 2.1 las herramientas y software a utilizar, así como la justificación de su elección.

SOFTWARE	JUSTIFICACIÓN
Java	1. Permite a través del uso de sesiones un manejo seguro de los usuarios. 2. Mediante los Servlets y jsp se obtienen datos y generan las páginas dinámicas para responder a las peticiones con información procedente de una base de datos. 3. La conexión a la BD y el intercambio de datos es posible con el driver adecuado que también está hecho con Java. 4. El API del lenguaje Java brinda las herramientas necesarias para desarrollar el proyecto en la mayoría de sus etapas sin recurrir a otros lenguajes para una función específica.
Eclipse	1. Es una aplicación ligera que no se instala ideal para equipos de desarrollo con bajos recursos. 2. Elegida la versión requerida y por su estructura de plugins instalables se tiene una aplicación con sólo lo necesario. 3. Puede asociarse al contenedor Apache Tomcat indispensable para el proyecto integrando un marco completo de desarrollo y pruebas para aplicaciones Web basadas en Servlets y jsp dentro del mismo IDE antes de ser llevado al equipo final. 4. Su editor de código soporta el HTML de las páginas que servirán de interfaz a los diferentes tipos de usuario eliminando la necesidad de Macromedia Dreamweaver para su diseño que si requiere pago de licencia.

Apache Tomcat	1. La generación de contenido dinámico y obtención de información de una base de datos puede realizarse con scripts incrustados en la página Web, lo cual puede considerarse riesgoso al mostrar datos privados del servidor o la base. Al emplear un contenedor de Servlets/JSP se pueden desarrollar aplicaciones y almacenarlas ahí en espera de ser solicitadas por la página Web, éstas se ejecutan del lado del servidor y pueden realizar las mismas funciones de forma más segura. 2. Al recibir la petición de un Servlet o jsp, la JVM responde creando la primera vez sólo una instancia y a partir de ésta se crean hilos para peticiones posteriores reduciendo el consumo de recursos en comparación con la tecnología CGI que responde a cada solicitud con un proceso.
Sybase ASE	1. El sistema y la base de datos estarán alojados en un servidor de USECAD el cual ya posee la licencia de éste sistema manejador de base de datos. 2. Las tablas requeridas con alumnos inscritos en teorías con laboratorio (+L) y de los propios laboratorios se encuentran bajo éste SMBD por lo cual será más sencilla su manipulación.

Tabla 1.1 Justificación de Software

2.5 Análisis de factibilidad

Teniendo presente la problemática existente y la justificación para el desarrollo de un proyecto de software integral, es indispensable realizar un estudio de factibilidad, el cual además de evitar una conclusión desastrosa y fallida del proyecto, determina en varios rubros si es posible su diseño y desarrollo evaluando aspectos operacionales, técnicos, económicos, los costos y beneficios, legales, entre otros.

Si la evaluación resulta positiva en cada uno se dice que el proyecto es viable permitiendo una mejor toma de decisiones bajo estudios reales de posibilidades.

Factibilidad operacional

Examina la posibilidad del cumplimiento del sistema en cuanto a sus funciones y objetivos al ejecutarse tal y como fue pensado, además de la aceptación por parte de todos los usuarios.

El proyecto cuenta con puntos a favor en este sentido, debido a que ciertas funciones serán semejantes a las integradas en sistemas conocidos por los usuarios desde hace algún tiempo, con los que ya están familiarizados como el caso de la asignación de calificaciones vía Web y la descarga de listas de alumnos inscritos, por lo cual se estima no habrá demasiados conflictos al migrar del método actual al nuevo sistema que ofrecerá una interfaz amigable y utilidades para reducir la dificultad y tiempo de los procesos.

Sin embargo, algunas funciones del sistema son nuevas y tal vez sean cuestionadas en un inicio al involucrar trámites no hechos hasta ahora como el llenado y entrega de formatos de corrección de calificación y registro de ASDRI, pero necesarios para una correcta administración en el Departamento de Computación al llevar una bitácora formal para estas acciones.

Lo que respecta a la administración y ejecución de funciones del sistema por el cliente (Personal del Departamento de Computación), éste deberá aprenderlo dejando atrás su ya conocido método pero se estima lograr su aceptación de forma rápida gracias a la comunicación continua y su inclusión durante todas las etapas del proyecto.

Con el fin de ampliar la aceptación de los usuarios y el cliente se pretende preparar materiales de apoyo a ser divulgados con suficiente tiempo para evitar confusiones en momentos críticos.

Considerando que el desarrollo e implementación de hace por petición del cliente bajo sus requisitos de funcionalidad, no existirán conflictos por su uso y por el contrario aumentará su productividad reflejada en menores tiempos y fallas. Sin embargo perderá cierto grado de control de información, pues anteriormente era almacenada en el equipo de cómputo personal del cliente y con la puesta en marcha del sistema ahora estará en el servidor de USECAD.

Finalmente, cabe destacar que todas las funciones solicitadas se consideran posibles de desarrollar sin implicar la creación de nuevas y riesgosas tecnologías.

Factibilidad técnica

Consistió en un análisis de la tecnología existente en el Departamento de Computación donde labora el cliente evaluando si era suficiente para implementar y soportar el sistema, arrojando datos negativos ante la ausencia del equipo indispensable para ello.

Tomando en cuenta los requerimientos técnicos se solicitó la cooperación de USECAD para proporcionar información vital para el funcionamiento del sistema, así como los recursos de hardware y software.

Dicha entidad cuenta con la experiencia, la tecnología y el personal calificados para el desarrollo y mantenimiento de aplicaciones de tipo Web con la capacidad suficiente de procesamiento, almacenamiento de información y atención a peticiones de usuarios.

El hecho de correr sobre el servidor de USECAD presenta 2 situaciones de ámbito técnico: la primera es la compatibilidad del sistema con el hardware y software del servidor; la segunda radica en una reducción significativa en el tamaño del proyecto dada la ubicación del sistema en el mismo sitio que la información requerida, evitando así el desarrollo de un módulo de interconexión entre sistemas incrementando la complejidad de programación y operación.

En cuanto al equipo de desarrollo, éste si se ubica en el Departamento de Computación y cumple con las especificaciones mínimas para la instalación de software.

Factibilidad económica

El análisis económico brinda un panorama de los recursos, costos y beneficios asociados al proyecto, la adquisición de equipo y la operación del sistema.

Aunque el proyecto carece de presupuesto para la compra de equipo, licencias de software o la contratación de personal para el desarrollo y operación, todos los aspectos se prevé sean cubiertos.

En primer lugar, el equipo servidor necesario será proporcionado por la USECAD, entidad responsable de otros servicios de cómputo, almacenamiento, publicación Web y sistemas institucionales, con lo cual se logra un beneficio mutuo al evitar la compra de equipo y la justificación de mayores recursos para la USECAD para el mantenimiento de sistemas.

Además, el equipo de desarrollo tampoco será adquirido al encontrarse uno disponible en el Departamento de Computación dedicado únicamente para el proyecto.

En lo que respecta al software, los de tipo propietario a emplear están instalados en el servidor y ya no es necesario pagar por su uso, los requeridos para el desarrollo y pruebas son de carácter libre evitando así la compra de software operativo, gestor de bases de datos, entornos de desarrollo y diseño, herramientas de administración de proyectos, entre otros.

Por otro lado tampoco es preciso el pago del personal por el desarrollo en vista de que es un proyecto de tesis ni la contratación adicional para su operación, la cual recae dentro de las actividades laborales del cliente.

El proyecto es viable financieramente al encontrarse los medios para cubrir todos los costos y ofrecer beneficios tanto para el cliente como para los usuarios, reduciendo la carga de trabajo en los equipos de escritorio del Departamento de Computación, las horas/hombre que eran destinadas a esta actividad, el riesgo de pérdida de información vital, así como los costos de operación y papelería.

Factibilidad legal

Análisis para determinar alguna infracción o responsabilidad legal en la que se incurra durante el proceso del proyecto, y a pesar de ser una tesis no se exime de cumplir reglamentos para ser aprobada e implementada.

Bajo el ámbito de la normatividad de la UNAM, en específico del Reglamento General de Exámenes de la Legislación Universitaria, se dicta en su Artículo 19 que la titulación en nivel licenciatura se expedirá al acreditar en su totalidad el plan de estudios, realizado el servicio social y cumplido con alguna de las opciones de titulación propuestas en el Artículo 20, de las cuales destaca en su Apartado "A" opción "a" la referida a la tesis y su evaluación.

Elegida esta forma de titulación y determinado el tema, objetivos, metas y alcances se registra en la Coordinación de Seminarios y Servicio Social de la Facultad de Ingeniería, donde se valora si cumple con el nivel académico y no afecta entidad o tema existente para finalmente ser aprobada por un Comité Evaluador de Exámenes Profesionales.

El presente proyecto superó esas etapas y fue aprobado constatando su viabilidad legal dentro del marco de la Legislación Universitaria.

2.6 Análisis de riesgo

El proceso del software no se encuentra aislado, también se ve afectado por su entorno y situaciones externas e inciertas llamadas riesgos que pueden provocar que el proyecto resulte mal como cambios en los requerimientos, tecnologías o personal.

Un riesgo es un problema potencial cuyo acontecimiento es impredecible, sin embargo debe ser identificado, clasificado, evaluada la probabilidad de que ocurra, estimar su impacto por nivel de daño, y planearse una estrategia con medidas enfocadas a evitarlo, contrarrestarlo cuando se presente y darle solución respondiendo de forma controlada y efectiva.

Se distinguen 3 categorías de acuerdo al grado de incertidumbre de ocurrencia y del grado de pérdidas:

- *Riesgos del proyecto.*

Afectan el presupuesto, la planeación y el personal, pueden ser derivados de problemas con la complejidad, tamaño del proyecto, los recursos, el cliente y los requisitos.

- *Riesgos técnicos.*

Ocurren ante dificultades en el diseño, implementación y mantenimiento debido a que el problema a resolver resultó más complejo de lo esperado en un principio o por factores de tecnología.

- *Riesgos de negocios.*

Afectan la viabilidad del software y se deben principalmente a los siguientes problemas:

- Riesgo de mercado. Nadie requiere el producto.
- Riesgo estratégico. El producto no corresponde con la estrategia comercial de la empresa.
- Riesgo de ventas. Construir un producto del cual no se tenga idea de cómo vender.
- Riesgos administrativos. Pérdida de apoyo de parte de la dirección o ejecutivos.
- Riesgos de presupuesto. Pérdida de recursos económicos o de personal.

Identificación del riesgo

Además de clasificarlos en categorías, un método recomendable para identificar y gestionar riesgos es realizar una lista de verificación de riesgos asociados en subcategorías genéricas:

- *Tamaño del producto (PS).*

Relacionados con el grado de certeza en la estimación del tamaño del software y base de datos, cantidad de módulos, archivos y código.

- *Impacto en el negocio (BU).*

Restricciones impuestas por la compañía o el gobierno, viabilidad del producto, costos debidos a retrasos y fallas, efecto en los ingresos.

- *Características del cliente (CU).*

Comunicación entre equipo de desarrollo y el cliente, conocimiento sobre cuestiones técnicas y del proceso de software por parte del cliente y su disposición para comentar requisitos y revisiones.

- *Definición del proceso (ES).*

Riesgos provocados por falta de conocimiento del proceso de software seleccionado, experiencias de implementación en otros proyectos o mal seguimiento documental de sus etapas. En la parte técnica refiere a la carencia de métodos o herramientas para el análisis, diseño, pruebas y medidas de calidad.

- *Entorno de desarrollo (DE).*

Los riesgos son causados por una mala herramienta o la ausencia de ésta para la gestión del proyecto o el desarrollo del software, por su falla en la integración con otras herramientas, una deficiente documentación y formación del equipo en su uso.

- *Tecnología a construir (TE).*

Riesgos latentes en la creación de nuevas tecnologías, en el uso de nuevos métodos de análisis y diseño, por la interacción del producto son otros sistemas de software y hardware no probados, y en la incertidumbre de conseguir la funcionalidad deseada.

- *Tamaño y experiencia de la plantilla (ST).*

Afines a la experiencia en proyectos y formación del equipo de trabajo, así como su disponibilidad a lo largo de todo el proceso.

Componentes de riesgo

- Rendimiento. Grado de incertidumbre respecto a si el producto cumplirá con los requisitos y el uso que se pretende darle.
- Costo. Grado de incertidumbre del mantenimiento del presupuesto del proyecto.
- Soporte. Grado de incertidumbre relacionado a la facilidad del software para adaptarse, corregirse y mejorarse.
- Planeación Temporal. Grado de incertidumbre para respetar el calendario del proyecto y fechas de entrega.

Proyección del riesgo

Una vez definidas las categorías y la lista de comprobación de riesgos se procedió a medir la probabilidad de que éstos ocurran, así como el impacto sobre el proyecto y los problemas que ocasionen.

La estimación del riesgo partió de cuatro actividades:

1. Establecer la probabilidad del riesgo.
2. Definir las consecuencias del riesgo.
3. Estimar el impacto del riesgo en el proyecto y en el producto a partir de la naturaleza, porcentaje de afectación y el tiempo que dure el problema.
4. Apuntar la exactitud general de la proyección del riesgo para no dejar lugar a confusiones.

Posteriormente se decidió proyectar todos los riesgos pensados sin importar la etapa donde podrían presentarse a través de una tabla de riesgos que se observa en la tabla 2.2.

RIESGOS	CATEGORÍA	PROBABILIDAD	IMPACTO
Dado que el proyecto es una tesis el personal está limitado.	ST	80	2
Falta de conocimiento en las herramientas, lenguajes de programación, bases de datos y sistemas operativos indispensables para el proyecto.	ST	70	2
Las tareas no técnicas como la redacción de capítulos y documentación necesitan más tiempo.	BU	70	2
La fecha final ha cambiado sin considerar si los recursos disponibles son suficientes ni el avance hasta el momento.	BU	60	1
La instalación en el entorno final puede causar fallas que no ocurrieron en el entorno de pruebas.	TE	60	2
El periodo de familiarización con las herramientas de software y los lenguajes de programación toma más tiempo del estimado.	BU	60	2
El cliente (Depto. Comp.) añade requisitos que no son viables o provocan cambios en el diseño.	PS	60	3
Se necesita regresar a una versión anterior más estable pero ya no está disponible.	ES	60	3
Falta de compromiso reduce el rendimiento.	ST	60	3
Los recursos de hardware ya no son suficientes para sustentar el desarrollo.	TE	50	2
El diseño es poco flexible para adaptarse al cambio de requisitos o la corrección de errores.	ES	40	2
La integración de los módulos es compleja o provoca fallas.	ES	40	2
El requisito de compatibilidad con el SMBD Sybase ASE implica cambios en el diseño.	TE	40	3
Faltaron especificaciones escritas de requisitos, diseño y código.	ES	40	3
No se llevó un seguimiento de las pruebas y errores para después darles solución.	ES	40	3
El entorno de desarrollo de programación elegido no cubre todos los aspectos necesarios.	DE	40	3
Los requisitos de hardware y software restrictivos retrasan el proceso de desarrollo.	BU	40	3
Existen retrasos en la entrega de prototipos y versiones.	BU	40	3
Las tablas de listas proporcionadas por USECAD están en un formato no contemplado provocando rediseño de módulos o de la base de datos.	PS	30	2
LÍNEA DE CORTE			

El requisito de compatibilidad con el SO Solaris puede implicar nuevos módulos.	TE	30	3
No se detectaron riesgos críticos en la etapa de análisis y el darles solución demanda más tiempo.	ES	30	3
El usuario final (Profesor de Teoría) está en desacuerdo con la interfaz que presenta las calificaciones.	BU	30	4
La falta de uso de herramientas de planificación y seguimiento de actividades provoca demoras.	ES	30	4
El número de usuarios finales (Profesor de Laboratorio) que asignen calificación de forma simultánea es mayor al esperado.	PS	20	2
Existen conflictos con tablas existentes en el SMBD.	TE	20	2
La presión reduce la productividad.	BU	20	3
La entrega de la información solicitada a USECAD demora más de lo previsto.	BU	20	3
El modelo de desarrollo de software elegido no fue el más adecuado.	ES	20	3
Los trámites de entrega y aprobación tardan más de lo esperado.	BU	20	4
El diseño de la interfaz para asignación de calificaciones no es del agrado del usuario final (Profesor de Laboratorio).	BU	20	4

Tabla 2.2 Proyección de Riesgos

En la primera columna se colocaron los riesgos resultantes de revisar toda la lista de verificación, en la segunda se indicó su clasificación de acuerdo a las categorías, en la tercera su probabilidad estimada, en la cuarta el impacto del riesgo evaluado y promediado para los cuatro componentes de riesgo representado como: 1-catastrófico, 2-crítico, 3-marginal, 4-despreciable.

Completadas las cuatro columnas se ordenaron priorizando las de mayor probabilidad y de ahí en segundo nivel por su impacto. Posteriormente se marcó una línea de corte horizontal separando los riesgos dejando sobre la línea los de alta a moderada probabilidad para ser tratados mediante un plan que los reduzcan, supervisen o administren.

Capítulo 3

Diseño del sistema

Una vez establecidos los requerimientos del cliente en cuanto a lo que desea del sistema, se procede al modelo lógico desde el enfoque orientado a procesos, para lo cual se convierten las especificaciones textuales a una representación simbólica apoyándose en la metodología Yourdon/De Marco para la etapa de diseño en el ciclo de vida del desarrollo de software.

Este método posee dicho enfoque estructurado basado en el concepto de proceso, en el que se pretende además tener una visión general de las funciones del sistema, de los datos que serán procesados y sus transformaciones a lo largo de las fases del programa considerando también las entradas, salidas y la influencia de agentes externos con los que el sistema tenga relación.

El diseño del sistema sigue un esquema de varias fases que involucran en esencia diagramas de flujo de datos, diccionario de datos y especificación de procesos y se definen a continuación:

- *Aspecto Ambiental.*

Se determina la relación del sistema con el exterior representándolo en un diagrama de contexto que identifica sus entradas, salidas, interfaces entre el sistema y el ambiente como pueden ser instrumentos de adquisición de datos y elementos de control. Estas entidades externas o también llamadas terminales del sistema son las encargadas de producir o recibir la información usada en el sistema.

El sistema completo es simbolizado en un sólo proceso con una burbuja y se conecta por flechas con los datos de entrada/salida hacia las entidades externas ignorando todas las tareas internas atendiendo únicamente los eventos provocados por el exterior y la respuesta que genera el sistema para cada evento.

La notación del diagrama de contexto se emplea también para el diagrama de flujo de datos descrito más adelante.

- *Aspecto de Comportamiento.*

La información fluye en el sistema procesando elementos de software, hardware y humanos como entradas y produciendo salidas que serán tomadas a su vez como entradas por otro proceso.

Para representar este comportamiento interno del sistema se emplea el diagrama de flujo de datos (DFD) que parte del diagrama de contexto pero agrega el concepto de almacén de datos.

El DFD es una herramienta de modelado gráfico cuyo objetivo es detallar los procesos a ser realizados y su relación con los datos almacenados en el sistema y conectados a través de los flujos de datos

En la figura 3.1 se observan los símbolos base del DFD y posteriormente una descripción de los elementos a los que representan.

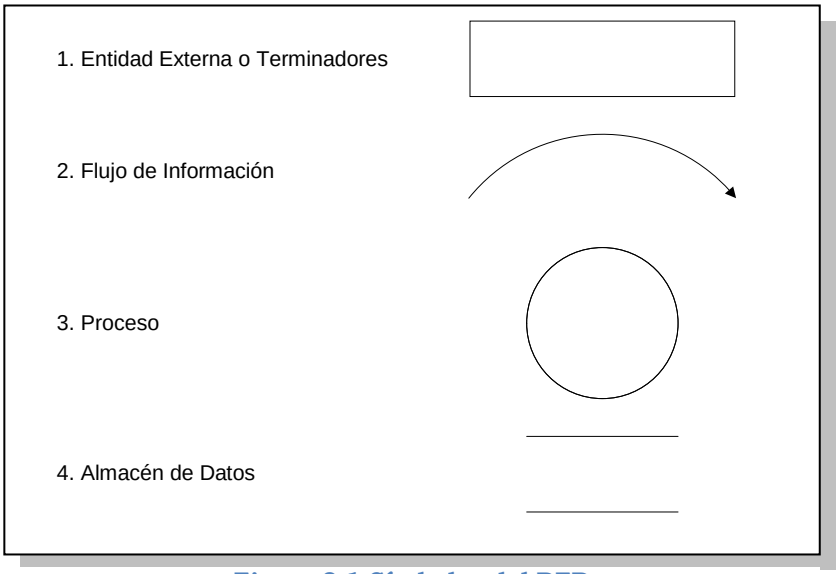


Figura 3.1 Símbolos del DFD

1. Elemento fuera del sistema que le provee de información o la recibe.
2. Información compuesta por un sólo elemento o por una estructura de datos que fluyen en una o ambas direcciones entre los procesos tomando los roles de entradas y salidas.
3. Son acciones de transformación, validación o distribución aplicadas a los datos y los modifican. Reciben el nombre en función de la acción en específico que realizan dentro del sistema como respuesta a un evento.
4. Es el archivo donde se agregan o extraen los datos para usarse temporalmente en forma de tablas de una base de datos.

A pesar de que el DFD proporciona una visión general de las funciones de sistema presenta algunas limitantes como el no ofrecer detalles de los datos involucrados ni la secuencia del procesamiento.

Para cubrir estos aspectos se recurre a herramientas como el diccionario de datos enfocado a ofrecer una descripción textual de la información, y al flujograma que muestra el orden del proceso del inicio al fin.

En el diccionario de datos cada elemento de información representado tanto en los flujos de datos como en los almacenes es organizado y detallado en forma precisa indicando características como nombre del elemento y su descripción.

Se puede implementar de forma manual, con un procesador de textos o con utilidades integradas en sistemas manejadores de bases de datos.

- *Aspecto de Información.*

Refleja la persistencia de la información a través del diagrama entidad – relación.

Los datos procesados por el sistema son empleados por los flujos del DFD, descritos en el diccionario y finalmente trasladados a una base de datos, pero es necesario además definir esos datos como entidades y las relaciones que existan entre ellas.

Este aspecto proporciona una visión más amplia y estructurada de la información al señalar las llaves primarias y foráneas indispensables para establecer las relaciones por las cuales los registros pueden ser procesados y consultados.

El uso de diagramas es parte fundamental en el diseño de sistemas al considerarse la transición hacia la implementación del código, también son elementos muy útiles en la posterior documentación y mantenimiento del sistema para el propio desarrollador u otras personas ajenas al proceso inicial al proporcionar una representación detallada de las funcionalidades del sistema facilitando las tareas de detección y corrección de fallas.

3.1 Diagramas de procesos

El sistema parte de un diagrama de contexto que se observa en la figura 3.2 donde se representa al sistema, los dispositivos externos como la impresora y los usuarios interactuando por los flujos de datos.

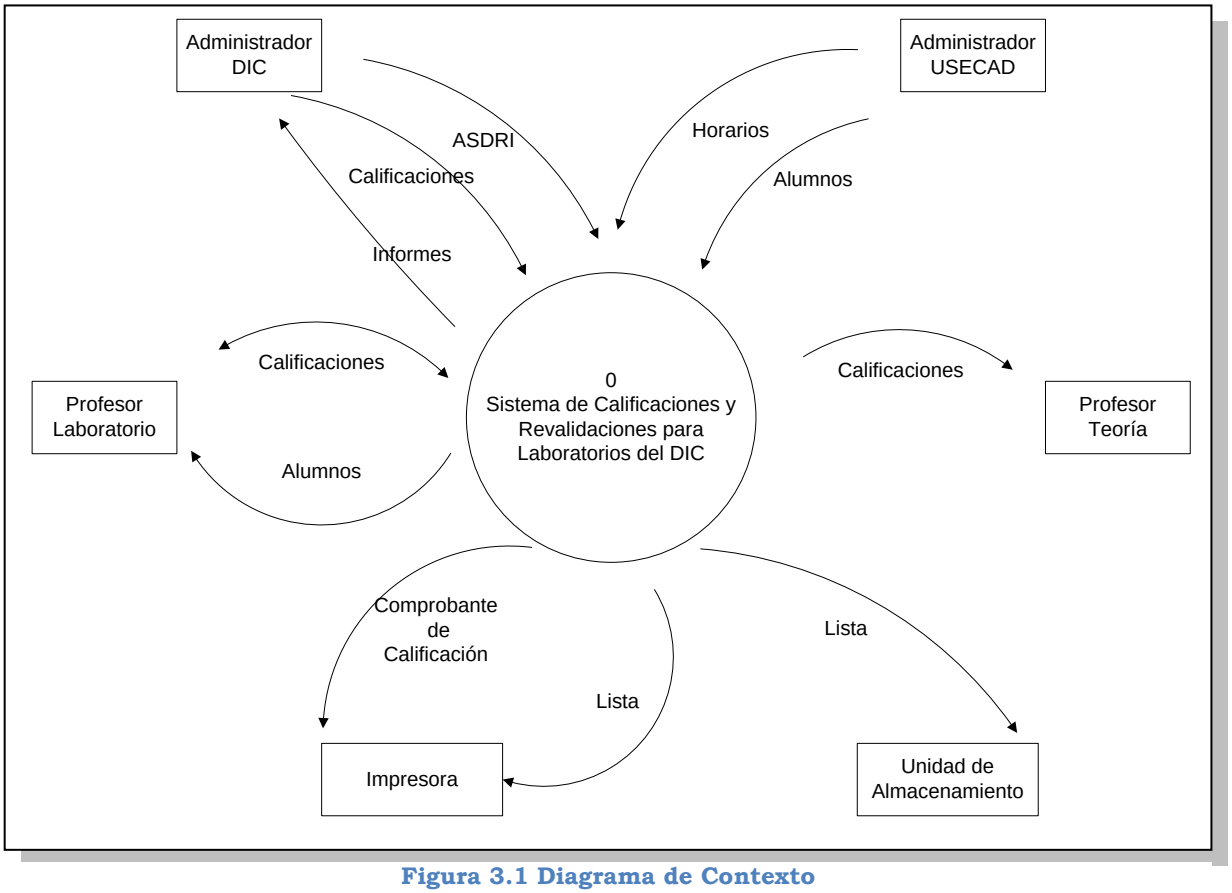


Figura 3.1 Diagrama de Contexto

El DFD puede descomponerse en subsistemas por niveles donde el número 0 es el diagrama de contexto, el nivel 1 integra todos los procesos que describen al proceso principal con una numeración que los distinga y finalmente, el nivel 2 o diagrama de expansión donde se detallan procesos provenientes del nivel anterior.

En la figura 3.3 se observa el Diagrama de Flujo de Datos de nivel 1 que contempla las funciones del sistema.

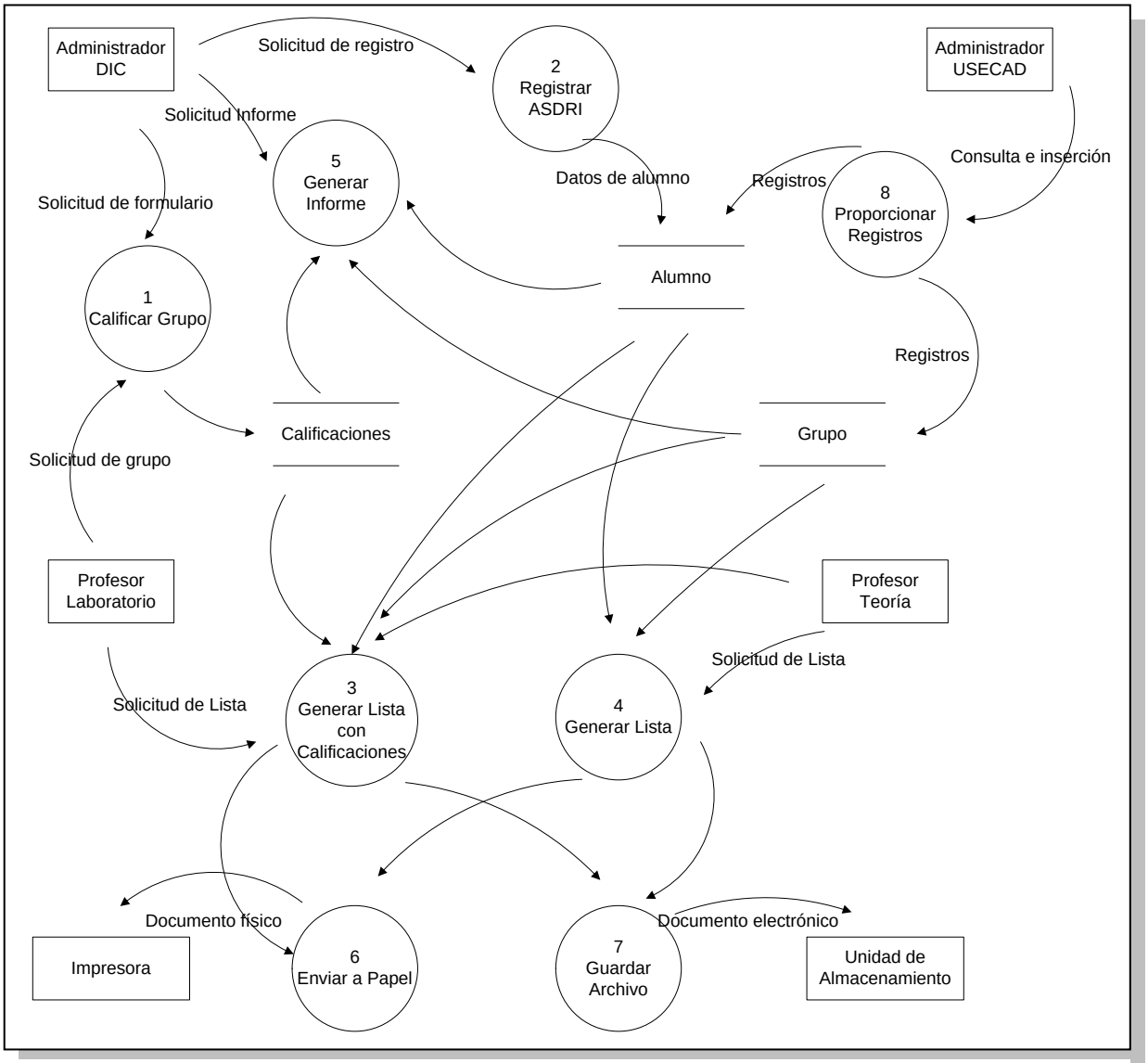


Figura 3.3 DFD Nivel 1

En la figura 3.4 se observa el nivel 2 de DFD donde se expande el proceso dedicado a la calificación de laboratorio con las sub-funciones que lo integran.

Este módulo es un tanto complejo por los procesos internos que modifican los flujos de datos para finalmente asignar una calificación tanto en forma individual como por todo un grupo de laboratorio.

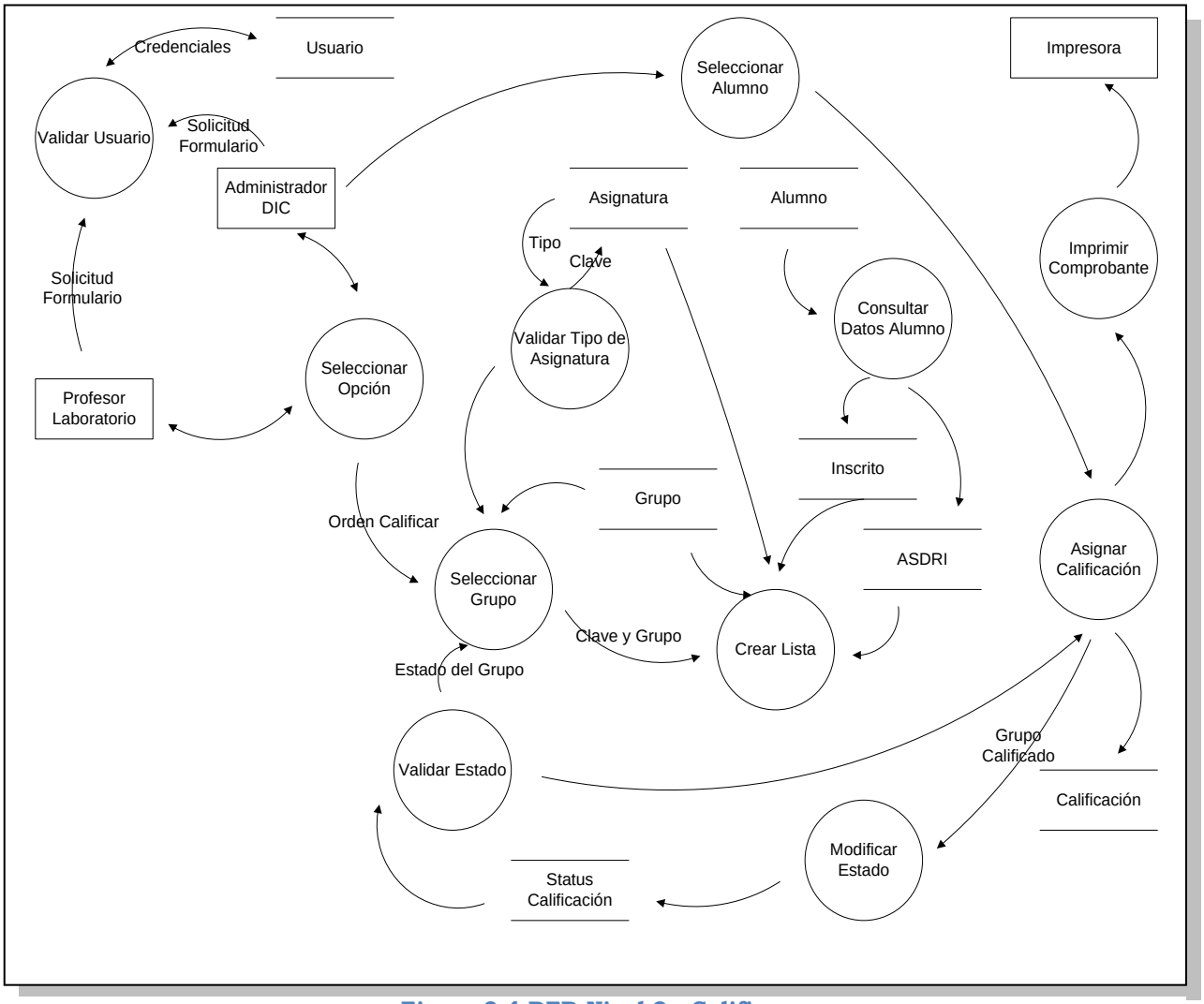


Figura 3.4 DFD Nivel 2 - Calificar

En la figura 3.5 se descompone el proceso de generación de listas que produce por grupo un documento electrónico o en papel con los alumnos inscritos, los alumnos sin derecho a reinscripción (ASDRI) y las calificaciones obtenidas en laboratorio para esa asignatura de teoría considerando todas las encontradas en semestres previos por si realiza el trámite de revalidación o la asignada en el presente semestre.

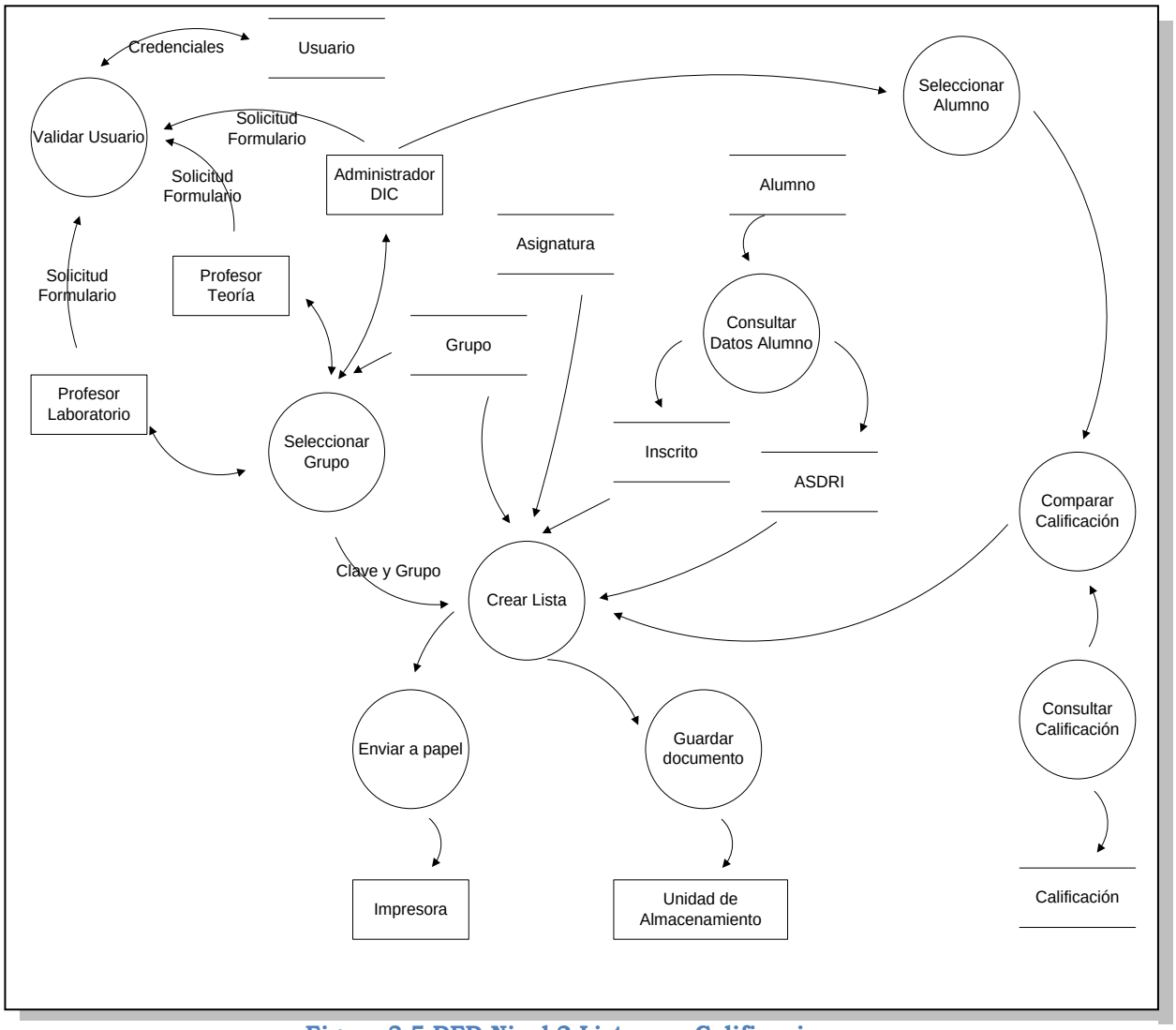


Figura 3.5 DFD Nivel 2 Lista con Calificaciones

El camino a seguir por el proceso dedicado al registro de ASDRI se observa en la figura 3.6 llevado a cabo por el usuario Administrador a partir de un formato impreso entregado por el alumno con los datos solicitados.

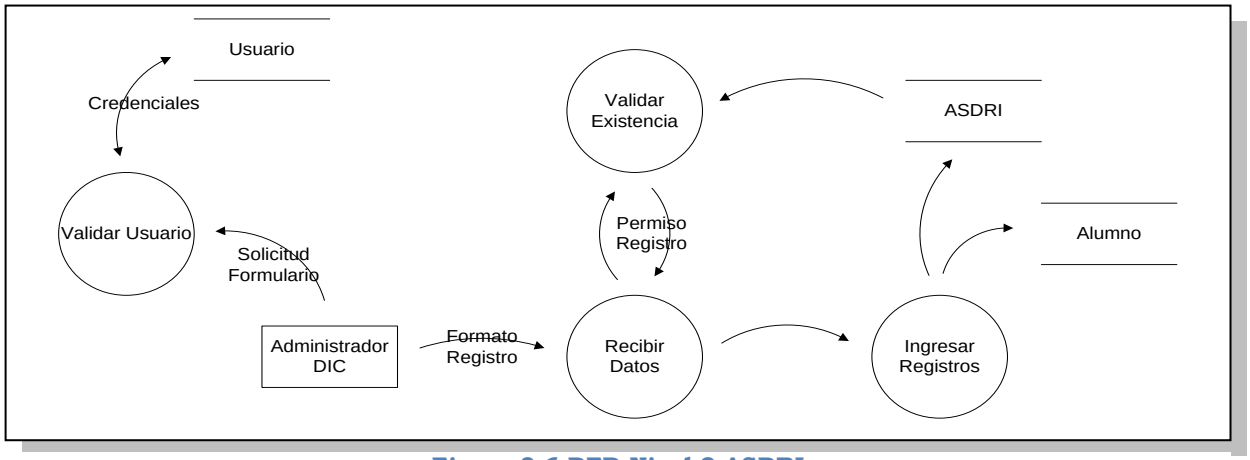


Figura 3.6 DFD Nivel 2 ASDRI

En la figura 3.7 se observa el proceso necesario para generar la lista para el profesor de laboratorio que a diferencia de la proporcionada directamente por USECAD, ésta si incluye ASDRI resultado del proceso de registro.

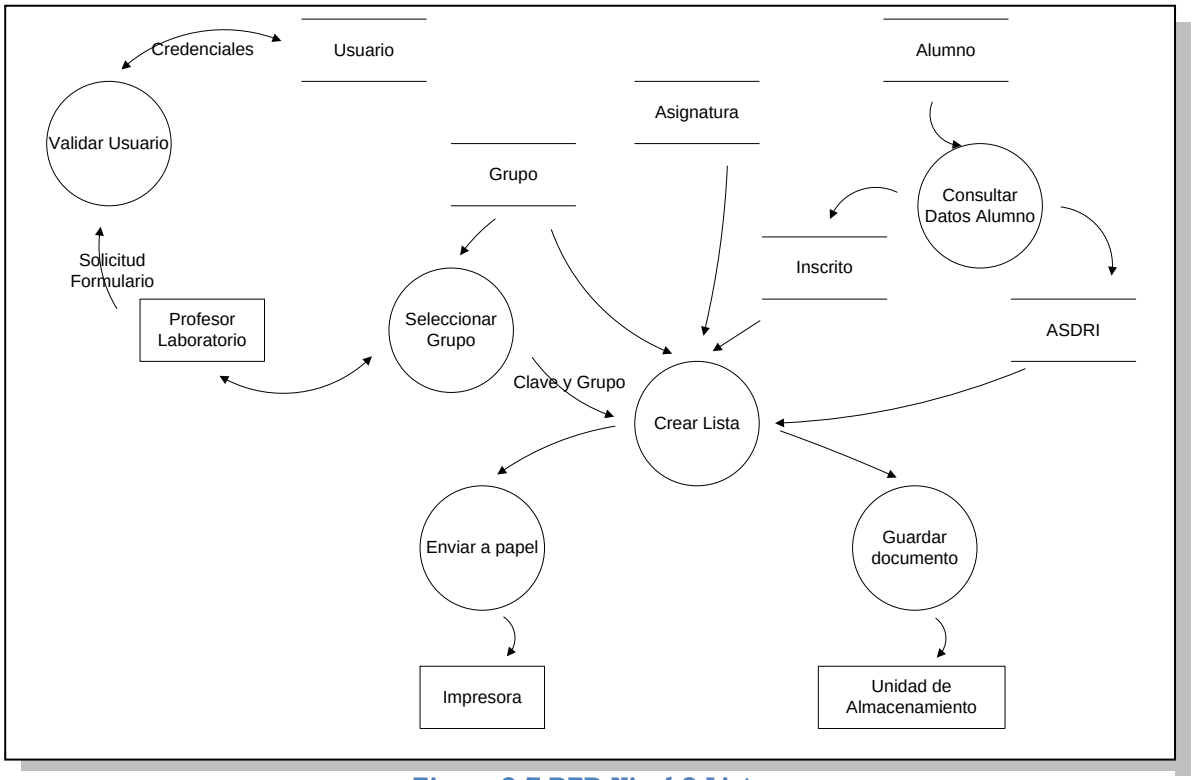


Figura 3.7 DFD Nivel 2 Lista

3.2 Diagramas de flujo

El diagrama de flujo o flujograma es la herramienta encargada de representar un proceso desde un inicio a un fin describiendo los detalles procedimentales en cada etapa, los usuarios involucrados y las decisiones que deben ser tomadas. También identifica el flujo de navegación de un sitio Web o de una aplicación. En la figura 3.8 se observa el diagrama de flujo general del sistema.

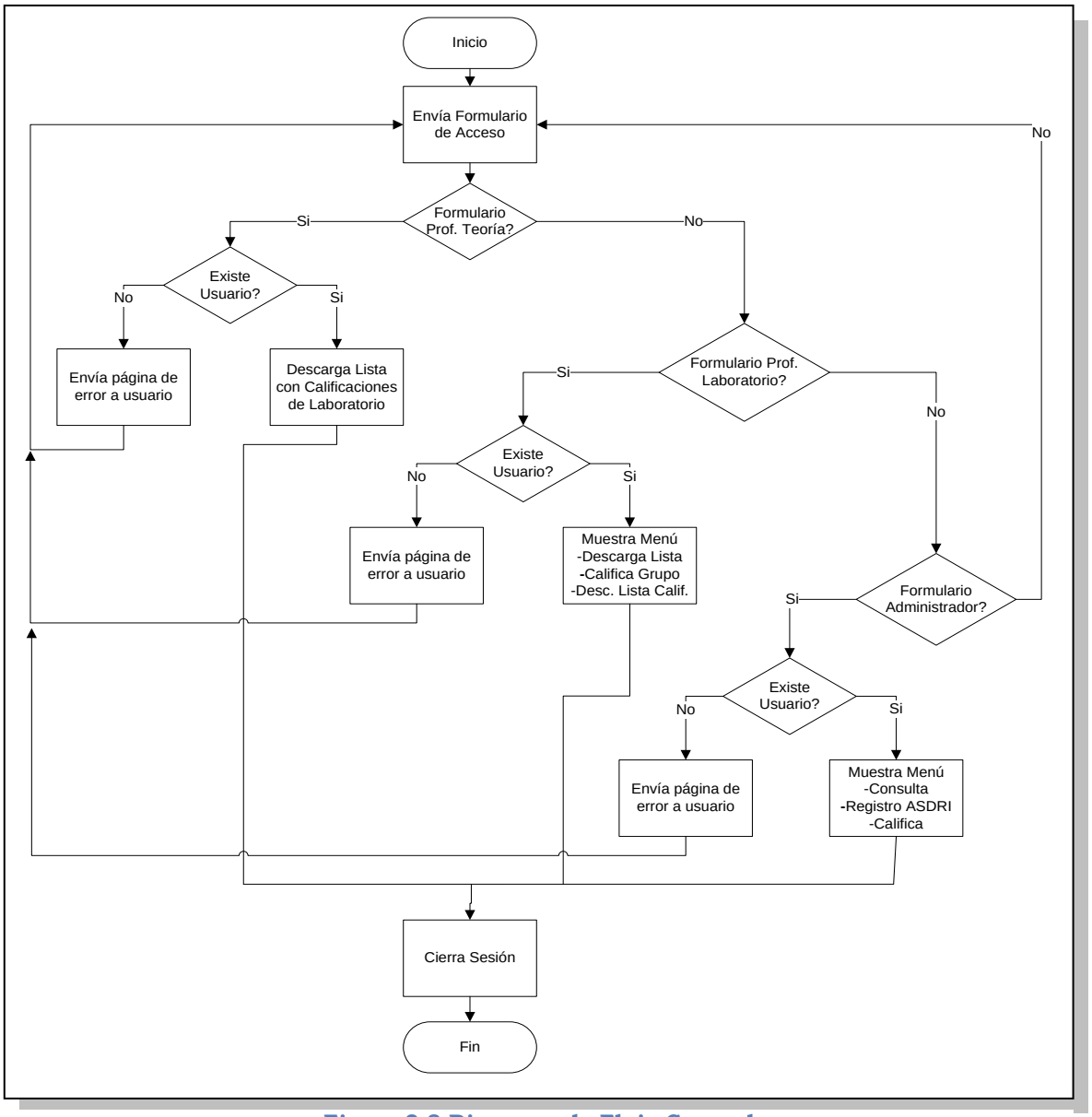


Figura 3.8 Diagrama de Flujo General

En la figura 3.9 se muestran los pasos para calificar un grupo.

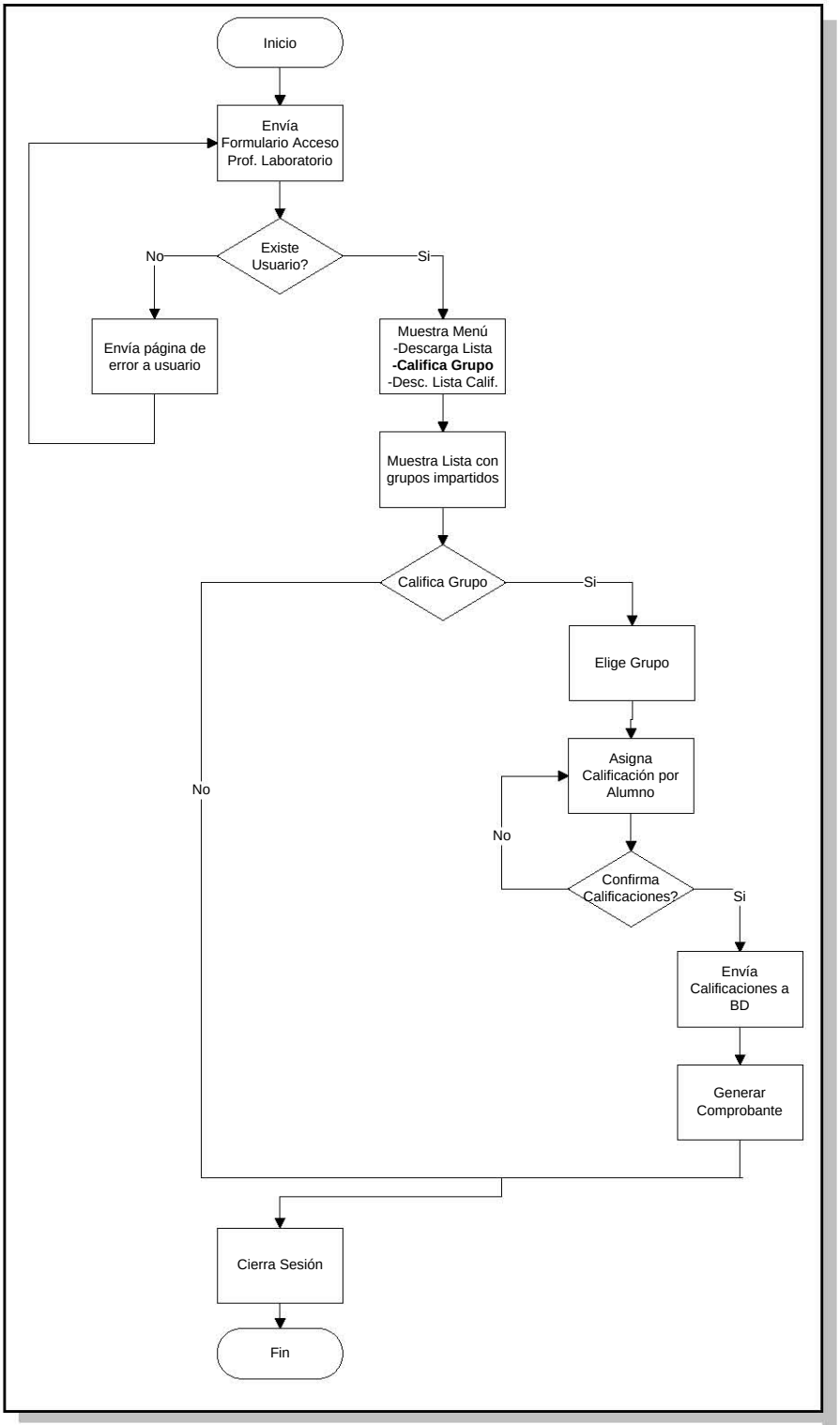


Figura 3.9 Diagrama de Flujo para Calificar

La secuencia para generar la lista con calificaciones se observa en la Fig. 3.10.

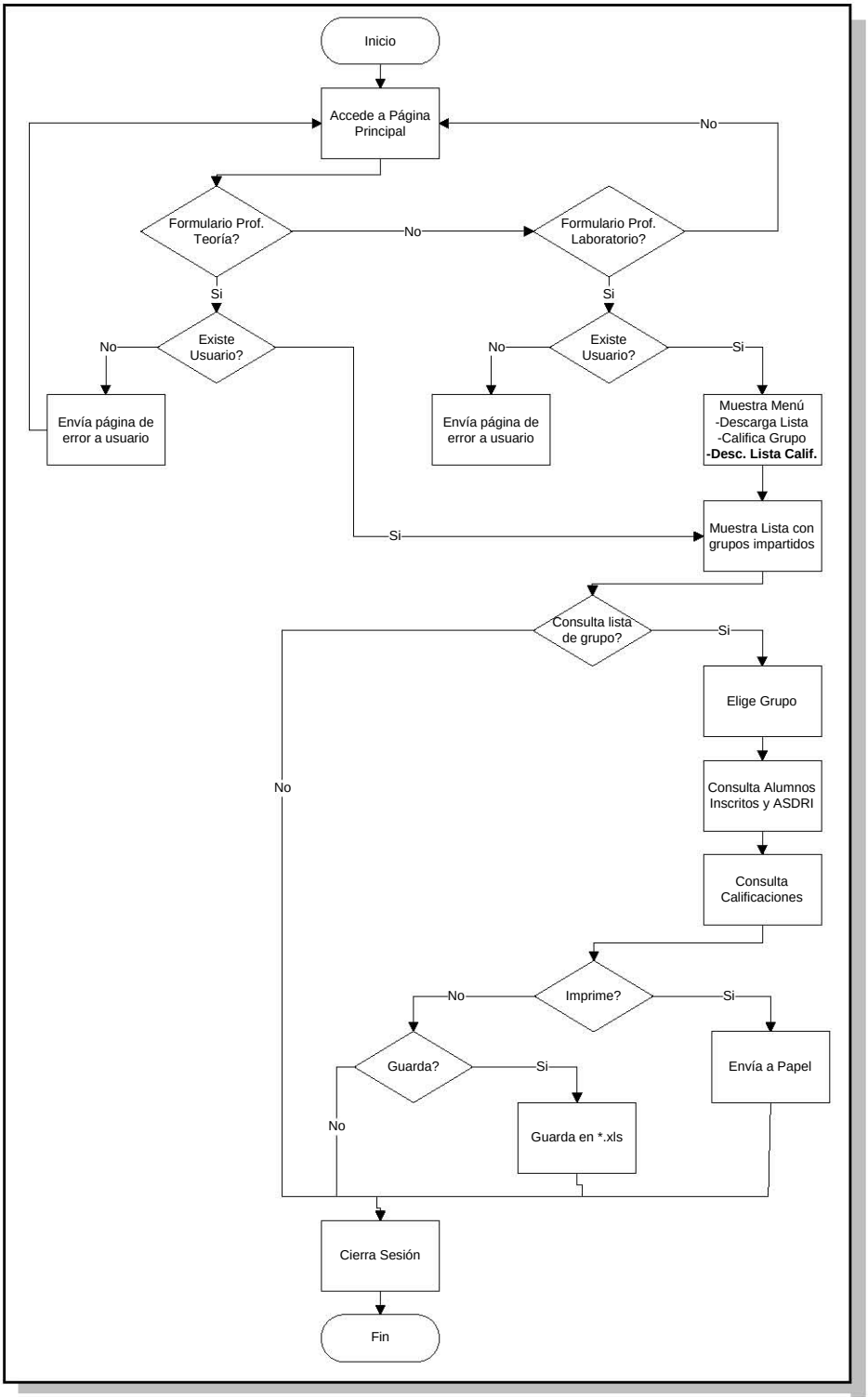


Figura 3.10 Diagrama de Flujo para Lista con Calificaciones

En la figura 3.11 se observa el proceso para el registro de ASDRI.

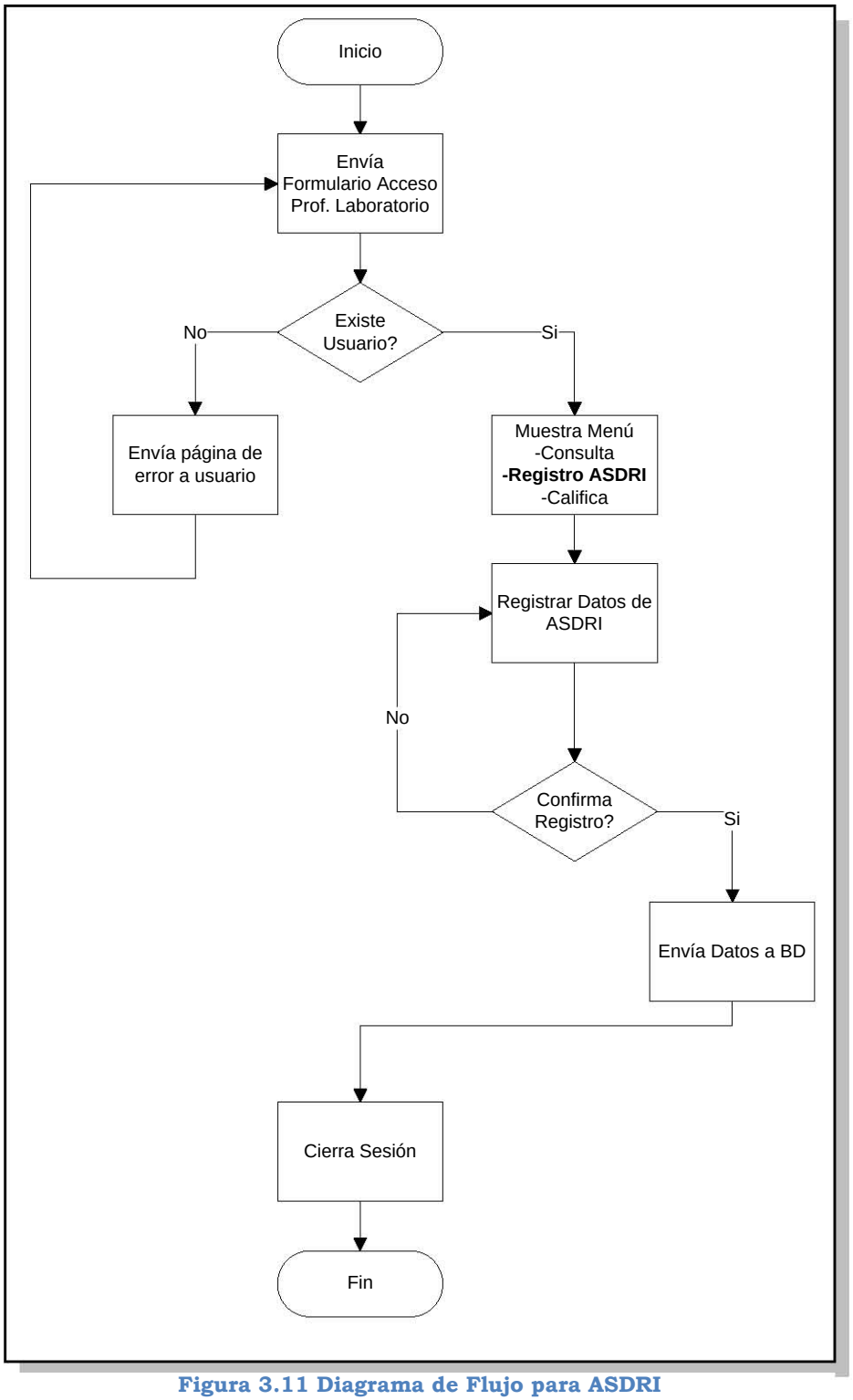


Figura 3.11 Diagrama de Flujo para ASDRI

El procedimiento para generar la lista de alumnos y ASDRI se ve en la Fig.312.

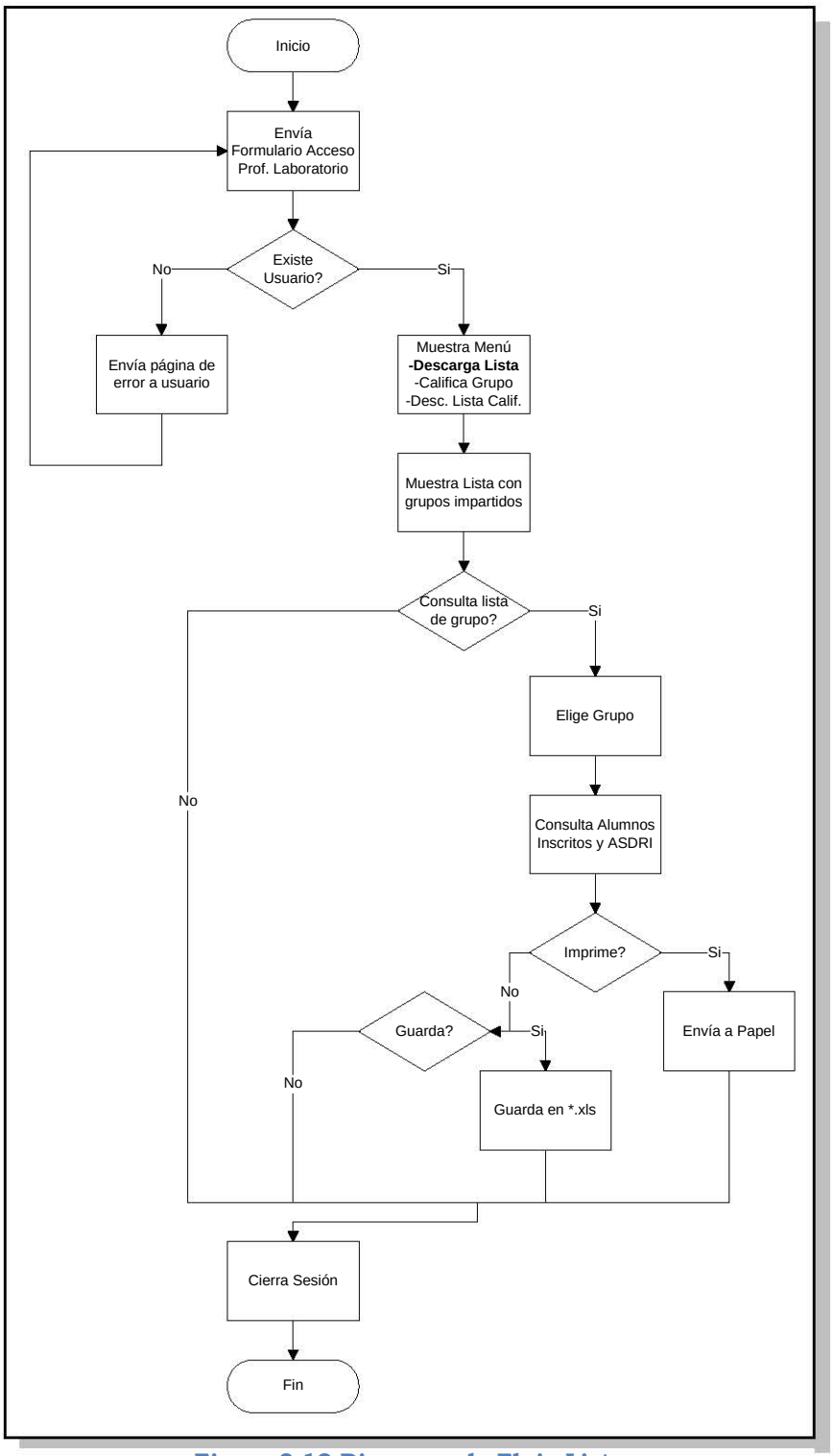


Figura 3.12 Diagrama de Flujo Lista

3.3 Diagrama entidad – relación

El sistema propuesto integra un conjunto de aplicaciones y páginas Web que tienen acceso a información de asignaturas, grupos y profesores para ser agregada, modifica o eliminada. Estos entes reales pueden ser representados mediante un modelo lógico y después en un modelo físico con una serie de tablas, registros y campos almacenados en una base de datos con una tecnología específica definiendo también las relaciones que existen entre ellas como se observa en la figura 3.13.

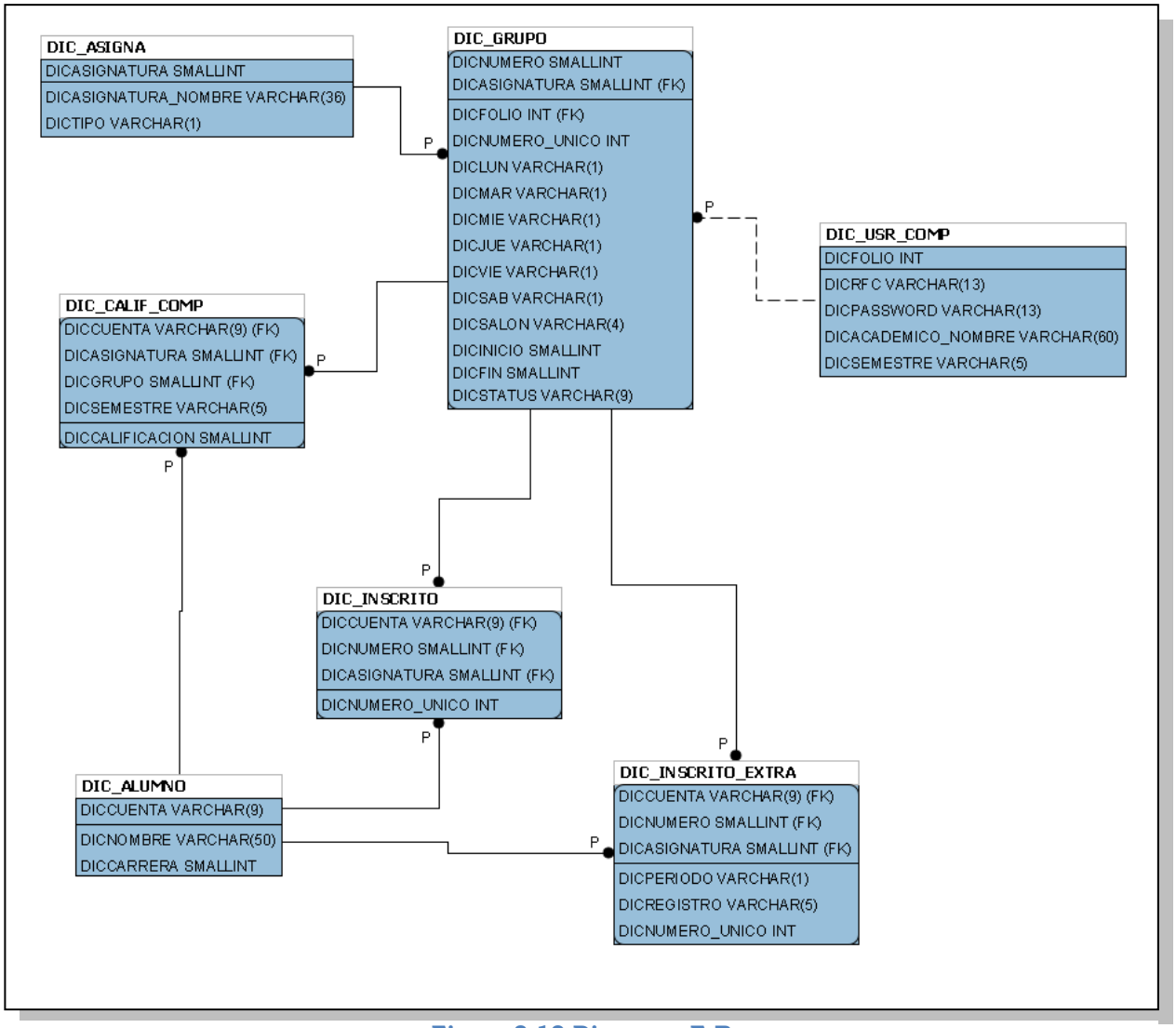


Figura 3.13 Diagrama E-R

3.4 Diccionario de datos

El diccionario de datos es la definición detallada de los elementos de datos y objetos de datos empleados por el sistema, refleja características como el nombre del elemento, tipo de dato para almacenamiento, si es que recibe datos nulos o es indispensable que contenga un valor, y si presenta la condición de fungir como llave primaria, foránea o única.

Estas características se muestran en las tablas 3.1 a 3.7 que corresponden a las tablas procedentes del diagrama entidad relación a ser implementadas en la base de datos.

TABLA DIC_ALUMNO

Field	Null	Type	Key	Default	Descripción
DICCUENTA	NO	varchar(9)	PRI		Número de cuenta del Alumno
DICNOMBRE	NO	varchar(50)			Nombre del Alumno
DICCARRERA	NO	smallint(6)			Carrera de pertenencia del Alumno

Tabla 3.1

TABLA DIC_ASIGNA

Field	Null	Type	Key	Default	Descripción
DICASIGNATURA	NO	smallint(6)	PRI		Clave de Asignatura
DICASIGNATURA_NOMBRE	NO	varchar(36)			Nombre completo de la Asignatura
DICTIPO	NO	varchar(1)			Indica si la Asignatura es Teoría o Laboratorio

Tabla 3.2

TABLA DIC_CALIF_COMP					
Field	**Null**	**Type**	**Key**	**Default**	**Descripción**
DICCUENTA	NO	varchar(9)	PRI		Número de cuenta del Alumno
DICASIGNATURA	NO	smallint(6)	PRI		Clave de Asignatura
DICGRUPO	NO	smallint(6)	PRI		Número de Grupo de la Asignatura
DICSEMESTRE	NO	varchar(5)	PRI		Año y número de semestre cuando alumno recibió calificación
DICCALIFICACION	NO	smallint(6)			Número de Calificación obtenida

Tabla 3.3

TABLA DIC_GRUPO					
Field	**Null**	**Type**	**Key**	**Default**	**Descripción**
DICNUMERO	NO	smallint(6)	PRI		Número de Grupo de la Asignatura
DICASIGNATURA	NO	smallint(6)	PRI		Clave de Asignatura
DICFOLIO	NO	int(11)	MUL		Número de registro del Profesor en la UNAM
DICNUMERO_UNICO	NO	int(11)			Identificador numérico del grupo
DICLUN	YES	varchar(1)			Lunes
DICMAR	YES	varchar(1)			Martes
DICMIE	YES	varchar(1)			Miércoles

DICJUE	YES	varchar(1)			Jueves
DICVIE	YES	varchar(1)			Viernes
DICSAB	YES	varchar(1)			Sábado
DICSALON	NO	varchar(4)			Número de salón donde se imparte la clase
DICINICIO	NO	smallint(6)			Hora de Inicio de la clase
DICFIN	NO	smallint(6)			Hora de Fin de la clase
DICSTATUS	NO	varchar(9)		FALTA	Indica si el Grupo ya fue o no Calificado

Tabla 3.4

TABLA DIC_INSCRITO

Field	Null	Type	Key	Default	Descripción
DICCUENTA	NO	varchar(9)	PRI		Número de cuenta del Alumno
DICNUMERO	NO	smallint(6)	PRI		Número de Grupo de la Asignatura
DICASIGNATURA	NO	smallint(6)	PRI		Clave de Asignatura
DICNUMERO_UNICO	NO	int(11)			Identificador numérico del grupo

Tabla 3.5

TABLA DIC_INSCRITO_EXTRA					
Field	**Null**	**Type**	**Key**	**Default**	**Descripción**
DICCUENTA	NO	varchar(9)	PRI		_
DICNUMERO	NO	smallint(6)	PRI		_
DICASIGNATURA	NO	smallint(6)	PRI		_
DICPERIODO	NO	varchar(1)			_
DICREGISTRO	YES	varchar(5)			_
DICNUMERO_UNICO	NO	int(11)			_

Tabla 3.6

TABLA DIC_USR_COMP					
Field	**Null**	**Type**	**Key**	**Default**	**Extra**
DICRFC	NO	varchar(13)	UNI		
DICPASSWORD	NO	varchar(13)		pass	
DICFOLIO	NO	int(11)	PRI		
DICACADEMICO_NOMBRE	NO	varchar(60)			
DICSEMESTRE	NO	varchar(5)		2000	

Tabla 3.7

Capítulo 4

Desarrollo e implementación

El sistema propuesto tiene un ambiente de ejecución en Internet por lo cual debe ser implementado sobre un servidor Web y un Sistema Manejador de Bases de Datos (SMBD), ambos recursos serán proporcionados por USECAD para su puesta en marcha final.

Al seleccionarse java como lenguaje de programación se garantiza la compatibilidad del sistema sin importar la arquitectura ni el sistema operativo donde se ejecute, así el proyecto concluido puede ser migrado desde la PC de desarrollo local hacia el servidor SUN Solaris.

En cuanto a la persistencia de la información se parte desde el modelado de datos hacia las tareas de creación de tablas y consultas en un SMBD libre para posteriormente exportarse al manejador final.

4.1 Creación de tablas

Una vez analizado y modelado el flujo de información, procesado por el sistema en forma de datos recibidos y enviados para ser almacenados como objeto de consulta o modificación, se diseñó un diagrama entidad-relación y un diccionario de datos que después será transformado a un modelo físico en una base de datos. Para esta fase se empleará el paquete de desarrollo Web EasyPHP encargado de instalar en conjunto el servidor Web Apache, PHP, la base de datos MySQL y el software de administración PHPMyAdmin para pruebas locales y el software Toad for Sybase como cliente para Sybase.

Toad for Sybase será la herramienta a utilizar como cliente en la conexión con el sistema manejador de bases de datos Sybase de USECAD para la creación de las tablas, las relaciones entre ellas al asignar las llaves primarias y foráneas, las consultas de registros, la importación y exportación de datos.

Todas estas funciones son mostradas en una interfaz gráfica intuitiva y de fácil uso que reduce la necesidad de escribir scripts SQL para las mismas actividades. La interfaz principal y la de creación de tablas se observan en las figuras 4.1 y 4.2 respectivamente.

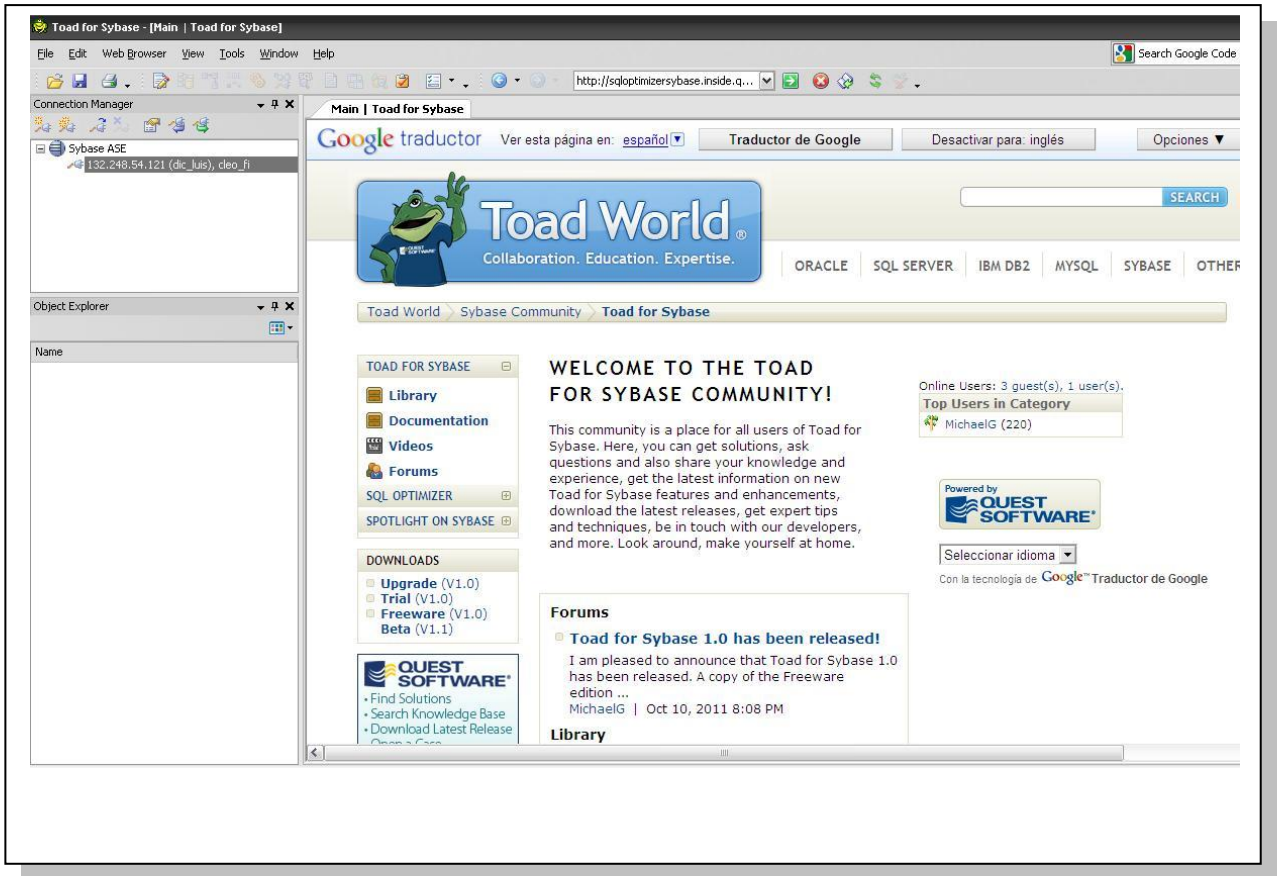


Figura 4.1 Interfaz Toad for Sybase

Los registros a conformar las tablas serán proveídos por USECAD, entidad responsable de administrar la información requerida relacionada a las asignaturas, grupos, alumnos y su condición de inscripción cada semestre, salvo lo concerniente al contenido de la tabla "DIC_CALIF_COMP".

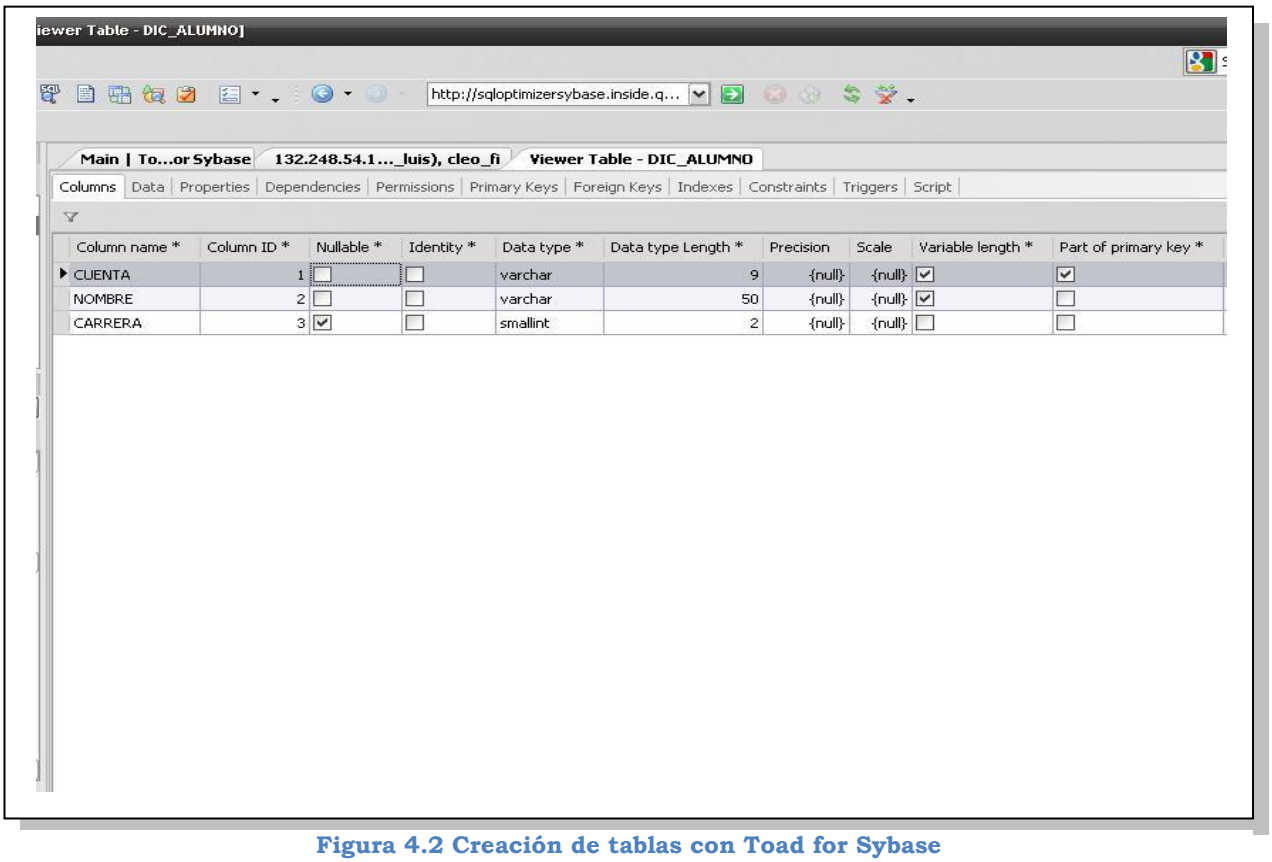


Figura 4.2 Creación de tablas con Toad for Sybase

Con el fin de cumplir el trámite de revalidación de calificaciones de laboratorios L+, todas las calificaciones asignadas desde el semestre 2006-2 deben ser almacenadas y estar disponibles para consulta por el sistema en caso de que un alumno no inscriba el laboratorio pero si la teoría por tenerlo acreditado en semestres previos.

Esta condición obliga a que la tabla "DIC_CALIF_COMP" se componga por todas las calificaciones guardadas pero que antes deben ser llevadas por un proceso de corrección, completado y depuración resultando en más de 32,000 registros a la fecha y se irán incrementando al paso de los semestres.

Los registros provienen originalmente de archivos de hoja de cálculo, uno por cada grupo de laboratorio y se encuentran de esta forma debido al requisito del Departamento de Ingeniería en Computación de enviar las calificaciones en un formato con este tipo de archivo. Los registros depurados se reúnen en un único archivo para ser importado por la herramienta de Toad for Sybase capaz

de soportar las extensiones .xls, .xlsx, .sql, entre otras y ser llevados finalmente a la tabla en la base de datos que los contendrá. Las opciones de importación se observan en la figura 4.3.

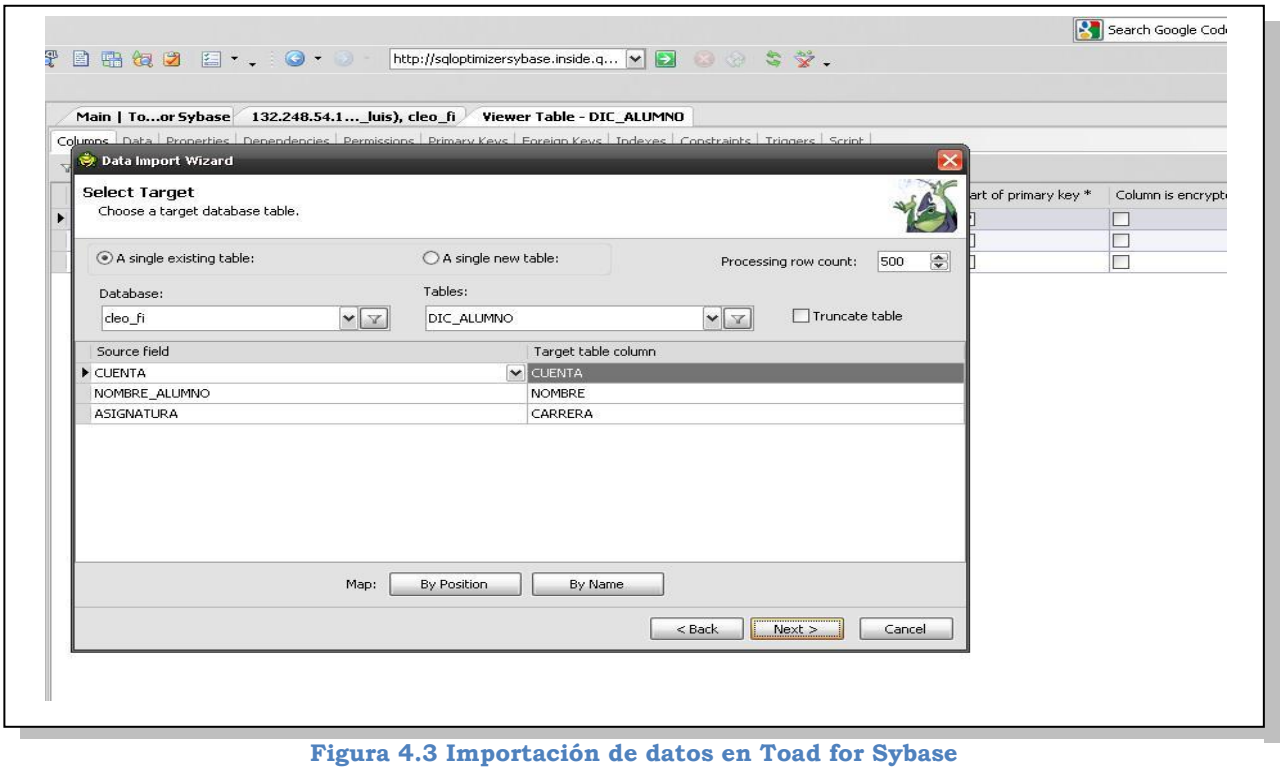


Figura 4.3 Importación de datos en Toad for Sybase

En lo respectivo a las tablas restantes, la información será copiada por el usuario Administrador de USECAD desde tablas almacenadas en su base de datos y son usadas por otros sistemas de la Facultad de Ingeniería. Un ejemplo de los scripts SQL creados para consultar los registros de sus tablas e insertarlos en las tablas del sistema puede verse en el Anexo 1.

Verificado el comportamiento esperado en las tablas y consultas sobre el SMBD libre, resta repetir el proceso de creación señalado en el diagrama entidad-relación en el SMBD Sybase con ayuda de algún cliente de conexión que ofrezca también una interfaz gráfica amigable.

4.2 Desarrollo de sistema

La siguiente fase al manejo de la información es la programación de los módulos del sistema basada en una estructura cliente-servidor, donde deben programarse funciones para ambos lados que cubran rubros de seguridad, acceso controlado a la información para agregar y modificar datos y por supuesto las interfaces de usuario.

Otra de las ventajas de java además de ser multiplataforma, es la amplitud de su campo de acción con aplicaciones de consola, gráficas y Web, siendo ésta ultima la naturaleza del sistema a desarrollar.

Las funciones ejecutadas en el servidor se basan en Servlets y las interfaces de usuario de carácter dinámico en HTML y JSP enriquecidas con JavaScript, lenguaje indispensable para efectos y comportamientos de muchos sitios en la actualidad.

Para la construcción de un proyecto de este tipo deben contemplarse situaciones para conseguir la rápida adaptación del usuario, algunas son:

- Los menús de navegación de todas las páginas del sitio deben estar bien ubicados y con una estructura común.
- Con la finalidad de recibir información correcta del usuario deben ofrecerse opciones visuales y evitar así la captura excesiva desde el teclado.
- Validar eventos y datos tanto del lado del servidor como del cliente para evitar errores o intrusiones.
- Prever comportamientos erróneos en el funcionamiento interno del sistema a causa de una falla en el acceso a la base de datos o por recibir datos no válidos, brindando al usuario una página de error personalizada con sugerencias para resolver el problema y su origen.
- Procurar que el diseño original del sitio se mantenga en todos los navegadores evitando restringir al usuario a alguno determinado.

El último punto es de vital importancia dado el número de navegadores disponibles, por lo cual se recurre a la biblioteca JavaScript JQUERY de distribución libre y código abierto orientada a simplificar funciones en menor código para manipular objetos de una página, hojas de estilos CSS, eventos del teclado y ratón, efectos y animaciones y por supuesto la compatibilidad con Mozilla Firefox, Internet Explorer, Safari, Opera y Google Chrome.

4.3 Implementación de módulos

El sistema está conformado por varios módulos implementados a partir de los diagramas de procesos y de flujo que corresponden a actividades como la calificación de laboratorios, registro de ASDRI y consultas de listas, todo vía Web y orientado a diferentes tipos de usuarios de acuerdo a la función, van de un profesor de teoría, de laboratorio, un administrador de la DIC o de USECAD hasta un ASDRI que únicamente desee descargar su formato de registro.

La página principal (figura 4.4) se compone por un menú lateral con los dos tipos de usuarios y una tercera opción para la descarga de formatos, accedida desde el portal de USECAD mediante un vínculo en la sección superior.

Figura 4.4 Página principal

Los usuarios profesor de teoría, profesor de laboratorio y administrador acceden a las opciones del sistema disponibles para cada uno al iniciar sesión desde una interfaz diferente con un formulario donde se introduce usuario y contraseña expuesta en las figuras 4.5 y 4.6.

Figura 4.5 Interfaz Profesor de Laboratorio

Figura 4.6 Interfaz Profesor de Teoría

Al presionar “Enviar”, el sistema validará la existencia del usuario y además si cumple la restricción de impartir ese tipo de asignatura. De no existir o no ser cierta la condición se redirige al usuario hacia una página de error con un código relacionado a la causa el cual puede comentar al administrador para mayor información como en la figura 4.7. El sistema prevé otros errores con su respectiva página y código.

Figura 4.7 Ejemplo de página de error

Con el fin de preservar la seguridad del sistema las interfaces destinadas al administrador DIC y de USECAD no serán expuestas, únicamente la interfaz con el menú de opciones mostrada en la figura 4.8.

La tercera opción de la página principal del sistema es un espacio dedicado a dos formatos descargables para ser llenados y entregados en el Departamento de Ingeniería en Computación que no requiere validación alguna de usuario por tanto la descarga es libre. Los dos archivos con extensión pdf son:

1. Formato de Registro de ASDRI de Laboratorio. (Véase figura 4.9)
2. Formato de Corrección de Calificación de Laboratorio. (Véase figura 4.10)

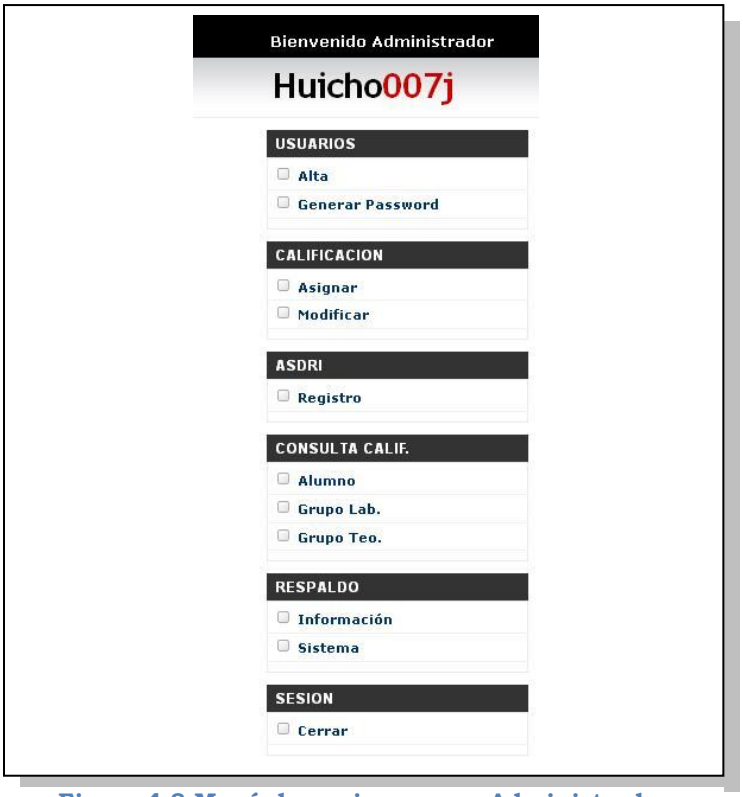


Figura 4.8 Menú de opciones para Administrador

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO Semestre
FACULTAD DE INGENIERÍA
SECRETARÍA DE SERVICIOS ACADÉMICOS

SOLICITUD DE REGISTRO A LOS LABORATORIOS PERTENECIENTES AL DEPARTAMENTO DE INGENIERÍA EN COMPUTACIÓN PARA ALUMNOS SIN DERECHO A REINSCRIPCIÓN

REGISTRO EN EL DEPARTAMENTO DE COMPUTACION: DEL------- AL -------

El correcto llenado de este formato y el registro en tiempo son responsabilidad del interesado.

Nombre: _____ Numero de Cuenta: ☐☐☐☐☐☐☐☐☐☐

Solicito mi registro en el laboratorio para alumno sin derecho a reinscripción, en la asignatura y grupo indicados a continuación, comprometiéndome a cumplir con todas las obligaciones que se enmarcan para la acreditación de la asignatura.

_____
Firma del Alumno

Nombre del laboratorio: _____ Clave del Laboratorio: ☐☐☐☐

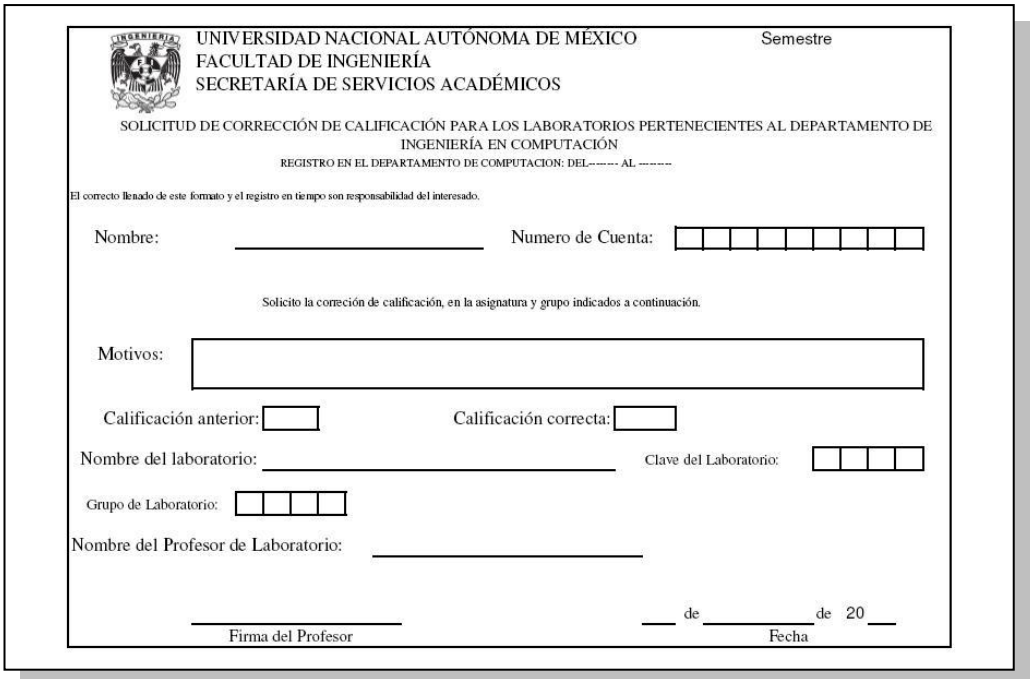
Grupo de Teoría: ☐☐☐☐ Grupo de Laboratorio: ☐☐☐☐

Nombre del Profesor de Laboratorio: _____

Acepto al alumno, cuyo nombre y número de cuenta se indican, en mi grupo durante el presente semestre.
Estimado profesor: en caso de aceptar al alumno en su grupo, favor de firmar este formato a más tardar el ---------- de -------

_____ ___ de _____ de 20___
Firma del Profesor Fecha

Figura 4.9 Formato de registro de ASDRI de laboratorio



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO Semestre
FACULTAD DE INGENIERÍA
SECRETARÍA DE SERVICIOS ACADÉMICOS

SOLICITUD DE CORRECCIÓN DE CALIFICACIÓN PARA LOS LABORATORIOS PERTENECIENTES AL DEPARTAMENTO DE INGENIERÍA EN COMPUTACIÓN

REGISTRO EN EL DEPARTAMENTO DE COMPUTACION: DEL------ AL -------

El correcto llenado de este formato y el registro en tiempo son responsabilidad del interesado.

Nombre: _____ Numero de Cuenta: | | | | | | | | | |

Solicito la correción de calificación, en la asignatura y grupo indicados a continuación.

Motivos: [ ]

Calificación anterior: [ ] Calificación correcta: [ ]

Nombre del laboratorio: _____ Clave del Laboratorio: [ ][ ][ ][ ]

Grupo de Laboratorio: [ ][ ][ ][ ]

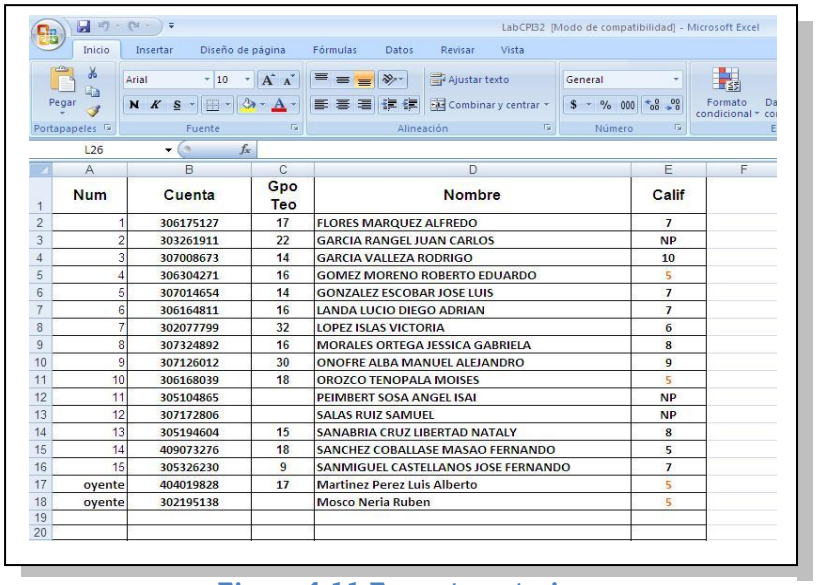
Nombre del Profesor de Laboratorio: _____

_____ ___ de _____ de 20 ___
Firma del Profesor Fecha

Figura 4.10 Formato de corrección de calificación de laboratorio

4.3.1 Calificación de laboratorio

Anteriormente las calificaciones de laboratorio eran enviadas en un formato con extensión xls mostrado en la figura 4.11.



LabCPI32 [Modo de compatibilidad] - Microsoft Excel

	A	B	C	D	E	F
1	Num	Cuenta	Gpo Teo	Nombre	Calif	
2	1	306175127	17	FLORES MARQUEZ ALFREDO	7	
3	2	303261911	22	GARCIA RANGEL JUAN CARLOS	NP	
4	3	307008673	14	GARCIA VALLEZA RODRIGO	10	
5	4	306304271	16	GOMEZ MORENO ROBERTO EDUARDO	5	
6	5	307014654	14	GONZALEZ ESCOBAR JOSE LUIS	7	
7	6	306164811	16	LANDA LUCIO DIEGO ADRIAN	7	
8	7	302077799	32	LOPEZ ISLAS VICTORIA	6	
9	8	307324892	16	MORALES ORTEGA JESSICA GABRIELA	8	
10	9	307126012	30	ONOFRE ALBA MANUEL ALEJANDRO	9	
11	10	306168039	18	OROZCO TENOPALA MOISES	5	
12	11	305104865		PEIMBERT SOSA ANGEL ISAI	NP	
13	12	307172806		SALAS RUIZ SAMUEL	NP	
14	13	305194604	15	SANABRIA CRUZ LIBERTAD NATALY	8	
15	14	409073276	18	SANCHEZ COBALLASE MASAO FERNANDO	5	
16	15	305326230	9	SANMIGUEL CASTELLANOS JOSE FERNANDO	7	
17	oyente	404019828	17	Martinez Perez Luis Alberto	5	
18	oyente	302195138		Mosco Neria Ruben	5	
19						
20						

Figura 4.11 Formato anterior

A pesar de su simplicidad era muy propenso a errores de captura y omisiones en los requisitos necesarios para la manera actual de recibir y distribuir calificaciones como el número de cuenta, el grupo de teoría donde está inscrito el alumno y su condición de inscrito o ASDRI.

El procedimiento queda sustituido por el módulo del sistema encargado de llevar al profesor de laboratorio por un proceso familiar para asignar calificaciones de forma ordenada y segura.

El primer paso es la validación del usuario desde la interfaz vista anteriormente, si el usuario existe y cumple la condición de impartir laboratorios se le presenta una nueva interfaz (figura 4.12) con sus grupos listados en una tabla y frente a cada uno 3 opciones de acciones posibles a realizar sobre el grupo más una columna adicional para indicar el estado de calificación del grupo con dos valores posibles: “Listo” y “Pendiente”.

Figura 4.12 Menú para Profesor de Laboratorio

Las 3 primeras columnas corresponden a las acciones:

1. Descargar lista con inscritos y ASDRI.
2. Calificar.
3. Descargar lista con calificaciones.

Cuando inicie el periodo del semestre reservado para calificar laboratorios todos los grupos tendrán como estado "Pendiente" y por tanto deshabilitada la opción de descarga de lista con calificaciones.

Los botones tradicionales del formulario fueron reemplazados por iconos relacionados a su acción y pueden ser vistos en la Tabla 4.1.

ÍCONO	ACCIÓN	ESTADO DEL BOTÓN
	Descargar lista	Habilitado
	Calificar	Habilitado
	Grupo calificado	Bloqueado
	Lista con calificaciones disponible	Habilitado
	Lista con calificaciones NO disponible	Bloqueado

Tabla 4.1 Íconos para botones

Al pulsar el ícono de la acción calificar el usuario es llevado a una nueva interfaz (figura 4.13) encabezada por una sección con la información de horario, salón y nombre de asignatura para el grupo seleccionado. Debajo encontrará su lista de inscritos y ASDRI proporcionando frente a cada alumno las posibles calificaciones (10, 9, 8, 7, 6 y NP) a asignar mediante botones de radio.

Universidad Nacional Autónoma de México

Facultad de Ingeniería

Departamento de Ing. en Comp.

Profesor						No. Prof	Tipo
LOPEZ GARCIA JORGE LUIS AYTE.						2	L
Asignatura						**Grupo**	**Clave**
LAB. DE PROG. AVANZADA Y MET. NUMER.						1	4312
Lunes	**Martes**	**Miércoles**	**Jueves**	**Viernes**	**Sábado**	**Salón**	**Inicio**	**Fin**
*						Q001	1100	1300

No.	Cuenta	Alumno	Calif	10	9	8	7	6	5	NP
1	308160613	ABARCA GARCIA ANGEL	NP	○	○	○	○	○	○	◉
2	307009869	ABURTO CEDILLO LUIS ALBERTO	NP	○	10	○	○	○	○	◉
3	308032484	ALVAREZ CORDOBA LUIS DAVID	NP	○	○	○	○	○	○	◉
4	308014949	ALVAREZ MULATO ERIK ARTURO	NP	○	○	○	○	○	○	◉
5	308174164	ARIAS SANTANA SERGIO	NP	○	○	○	○	○	○	◉
6	304113040	CASTILLO FLORES CARLOS	NP	○	○	○	○	○	○	◉
7	308062421	CASTILLO MENDEZ CARMEN ELIZABETH	NP	○	○	○	○	○	○	◉
8	308027802	CHAVEZ PADILLA PEDRO	NP	○	○	○	○	○	○	◉

Figura 4.13 Interfaz de calificación

Concluida esta actividad para todos los alumnos del grupo el profesor podrá presionar el botón de “Calificar” y después de confirmar la acción mediante una ventana emergente las calificaciones asignadas serán enviadas a la base de datos del sistema generando en el momento también un comprobante de finalización del proceso en formato pdf con la misma información del grupo y calificaciones para ser impreso.

La interfaz se modifica por último como respuesta al evento impidiendo la modificación o envío permitiendo únicamente al usuario cerrar sesión.

Un ejemplo del comprobante de calificación enviado al profesor se muestra en la figura 4.14.

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
FACULTAD DE INGENIERIA
DEPARTAMENTO DE INGENIERIA EN COMPUTACION

COMPROBANTE DE CALIFICACION

FECHA DE CALIFICACION: 09-11-2011

PROFESOR: LOPEZ GARCIA JORGE LUIS AYTE.	**CLAVE: 4312**	**SEM: 2012-1**
ASIGNATURA: LAB. DE PROG. AVANZADA Y MET. NUMER.	**GRUPO: 1**	***S11091753***

No.	CUENTA	NOMBRE DEL ALUMNO	CALIF.
1	308160613	ABARCA GARCIA ANGEL	10
2	307009869	ABURTO CEDILLO LUIS ALBERTO	9
3	308032484	ALVAREZ CORDOBA LUIS DAVID	8
4	308014949	ALVAREZ MULATO ERIK ARTURO	7
5	308174164	ARIAS SANTANA SERGIO	8
6	304113040	CASTILLO FLORES CARLOS	8
7	308062421	CASTILLO MENDEZ CARMEN ELIZABETH	7
8	308027802	CHAVEZ PADILLA PEDRO	9
9	306172102	CORTES JIMENEZ LUIS ENRIQUE	10
10	308188732	CORTEZ VELEZ MIGUEL ANGEL	8
11	308304521	DORANTES AROSTICO JOSE LUIS	6
12	308022656	ESQUIVEL VARGAS CARLOS ANDRES	9
13	305226989	GALVAN HERNANDEZ KARINA	8
14	307068444	GONZALEZ BENITEZ DAVID NATANAEL	7
15	308002447	GONZALEZ MOSQUEDA JOSE DE JESUS	5

Figura 4.14 Comprobante de calificación

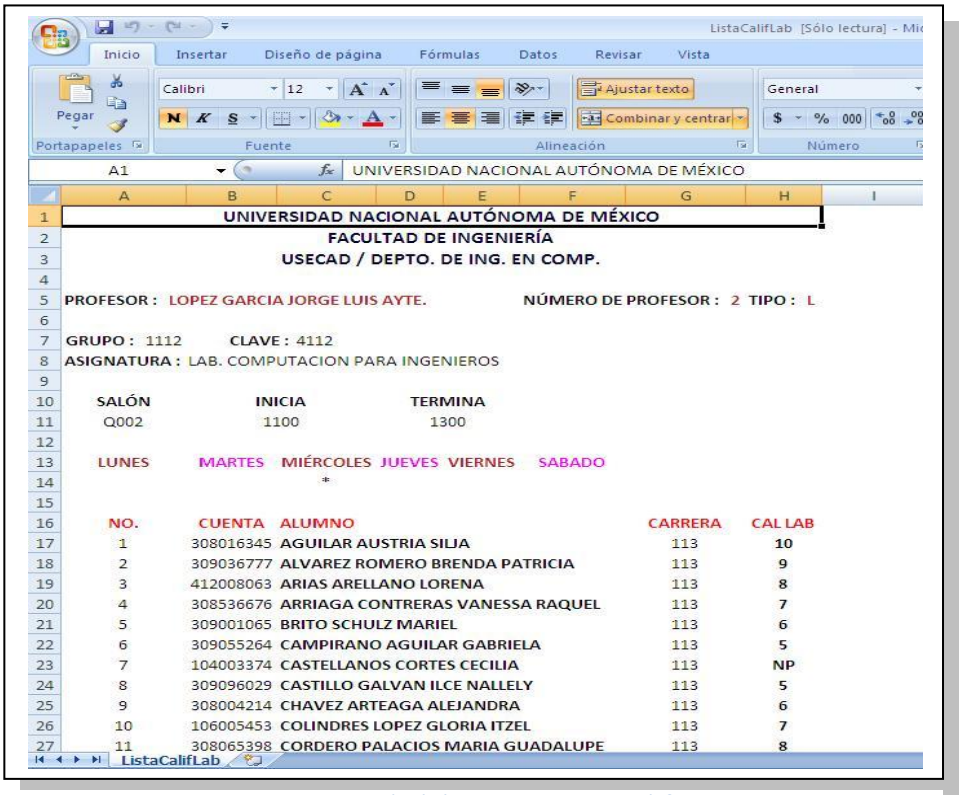
4.3.2 Registro de alumnos sin derecho a reinscripción de laboratorio

El proceso de registro de alumnos sin derecho a reinscripción (ASDRI) tiene como objetivo mantener un control de los alumnos de laboratorio en esta condición antes inexistente, pero además incorporarlos en la lista del profesor de laboratorio para su posterior descarga o calificación con lo que se garantiza su pertenencia al grupo y la fiabilidad de sus datos. El ASDRI podrá descargar el formato de registro disponible desde el portal, imprimirlo, anotar la información requerida y recibir la firma de consentimiento del profesor para después entregarlo en el Departamento de Ingeniería en Computación. Hecho ésto, el usuario Administrador DIC será el encargado de generar el registro en el sistema y almacenarlo a partir del formato.

4.3.3 Descarga de listas con calificaciones de laboratorio

Este módulo cubre 2 usuarios en vista de que tanto el profesor de teoría como el de laboratorio pueden descargar la lista de alumnos con su calificación obtenida en el laboratorio pero desde la propia interfaz de su tipo de usuario.

En el caso del profesor de laboratorio la acción de descarga se realiza desde la página con el menú de grupos y opciones ya mencionada anteriormente resultante de la validación correcta del usuario. Considerando un grupo ya calificado basta con dar click sobre el ícono de la tercer columna a lo que el sistema responderá con el cuadro de diálogo "Abrir/Guardar" del navegador para el archivo de hoja de cálculo a descargar. El documento como se observa en la figura 4.15 consta de los datos del grupo y varias columnas destinadas al nombre del alumno, número de cuenta, carrera y la calificación dada por él en el módulo de calificación de laboratorio.



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
USECAD / DEPTO. DE ING. EN COMP.

PROFESOR : LOPEZ GARCIA JORGE LUIS AYTE. NÚMERO DE PROFESOR : 2 TIPO : L

GRUPO : 1112 CLAVE : 4112
ASIGNATURA : LAB. COMPUTACION PARA INGENIEROS

SALÓN	INICIA	TERMINA
Q002	1100	1300

LUNES	MARTES	MIÉRCOLES	JUEVES	VIERNES	SABADO
		*			

NO.	CUENTA	ALUMNO	CARRERA	CAL LAB
1	308016345	AGUILAR AUSTRIA SILJA	113	10
2	309036777	ALVAREZ ROMERO BRENDA PATRICIA	113	9
3	412008063	ARIAS ARELLANO LORENA	113	8
4	308536676	ARRIAGA CONTRERAS VANESSA RAQUEL	113	7
5	309001065	BRITO SCHULZ MARIEL	113	6
6	309055264	CAMPIRANO AGUILAR GABRIELA	113	5
7	104003374	CASTELLANOS CORTES CECILIA	113	NP
8	309096029	CASTILLO GALVAN ILCE NALLELY	113	5
9	308004214	CHAVEZ ARTEAGA ALEJANDRA	113	6
10	106005453	COLINDRES LOPEZ GLORIA ITZEL	113	7
11	308065398	CORDERO PALACIOS MARIA GUADALUPE	113	8

Figura 4.15 Lista de laboratorio con calificaciones

Para el caso del profesor de teoría el proceso parte también de la validación del usuario y a continuación en una nueva interfaz se listan los grupos de teoría impartidos por el profesor en un combo desplegable junto con la clave, nombre de la asignatura y tipo (figura 4.16). Después de seleccionar el grupo deseado podrá presionar el botón "Descargar" y el sistema responderá también con el cuadro de diálogo para abrir o guardar el documento .xls diseñado de igual forma al de laboratorio, con la diferencia de que la calificación fue consecuencia de la búsqueda por parte del sistema y comparación alumno por alumno de sus calificaciones existentes para esa asignatura, tomada ya sea de la asignada por el profesor de laboratorio ese semestre o anteriores.

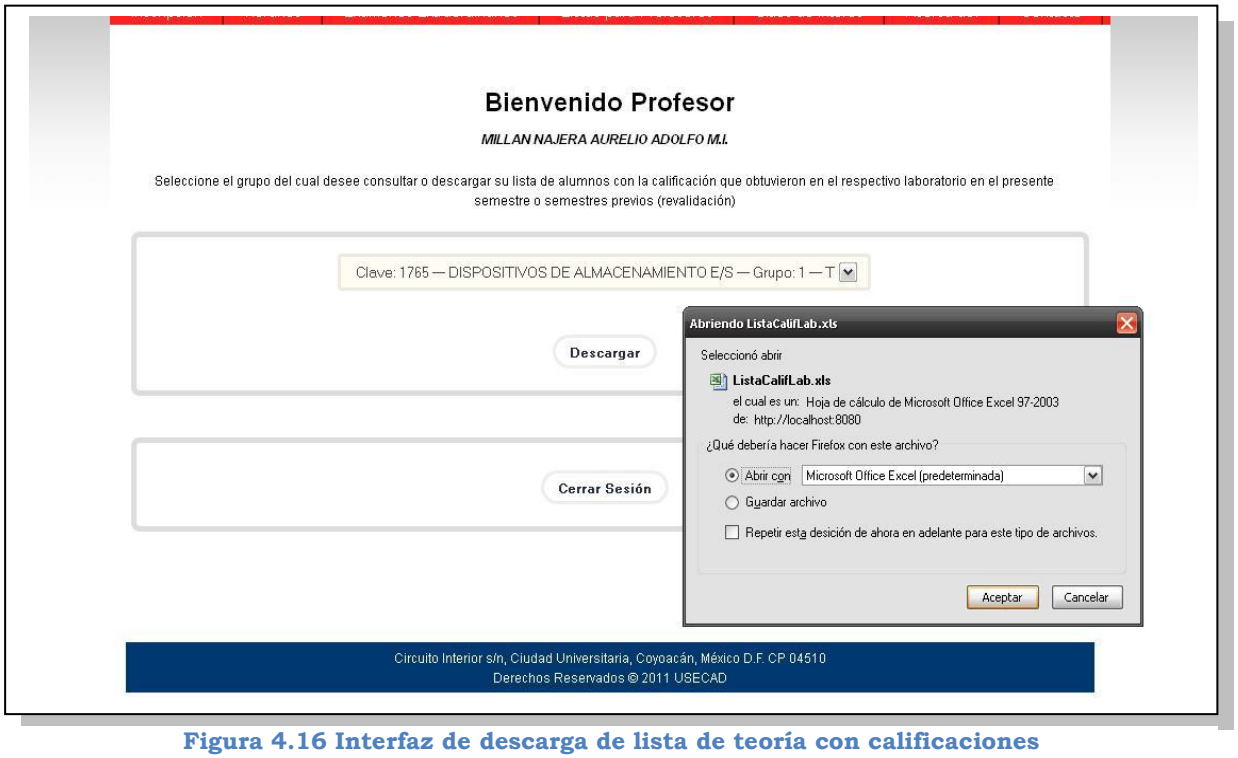


Figura 4.16 Interfaz de descarga de lista de teoría con calificaciones

El sistema mantiene la interfaz activa en espera de que el profesor desee descargar la lista de otro grupo evitando inicie nuevamente sesión o por el contrario puede finalizarla con el botón de cierre.

4.3.4 Descarga de listas con inscritos y alumnos sin derecho a reinscripción de laboratorio

Previa validación de la existencia del usuario y la condición de impartir laboratorios, el profesor tendrá disponible la tabla con grupos impartidos y acciones a realizar donde puede presionar el ícono de la primer columna correspondiente a la descarga y a lo cual el sistema responderá enviando el cuadro de diálogo "Abrir/Guardar" del navegador para un archivo .xls. De acuerdo al grupo seleccionado el contenido del documento de hoja de cálculo será la lista de alumnos inscritos y ASDRI registrados por el usuario Administrador DIC junto con los datos de horario y asignatura en la parte superior.

4.3.5 Respaldo de información

Módulo ejecutado por los usuarios Administrador y encargado de realizar copias de seguridad de la información del sistema constantemente a tablas dentro de la propia base de datos.

Este módulo es necesario para preservar datos vitales para el funcionamiento del sistema como las calificaciones de laboratorios en casos de intrusiones, fallas en el servidor o el manejador o simplemente tener respaldos al día de las tablas.

4.3.6 Respaldo de sistema

Consiste en realizar por parte del Administrador USECAD una copia periódica del proyecto montado sobre el servidor Apache Tomcat llevando todos los archivos del sistema como son páginas Web, imágenes, hojas de estilos, archivos de JavaScript y por supuesto los Servlets encargados del funcionamiento.

Las opciones de respaldo constituyen el mantenimiento preventivo del sistema indispensable en caso de fallas y sea necesaria su inmediata puesta en marcha teniendo todos los elementos necesarios para implementarse sobre el mismo equipo o en otro.

Conclusiones

El sistema desarrollado cumplió con los objetivos de forma que se obtuvo un software acorde a los requisitos y necesidades del cliente.

Cada uno de los módulos brinda los beneficios deseados a cada una de las categorías de usuarios con una interfaz amigable y sencilla para agilizar los procesos de calificación de laboratorios, revalidaciones y descarga de listas.

Además se consigue dejar atrás los errores cometidos con la metodología anterior en cuanto a la confiabilidad y rapidez de la información recibida con mejores medios de obtención y control.

Por otro lado se logra llevar un registro de los ASDRI de laboratorio para considerarlos en la lista de inscritos y en la evaluación, así como sucede en las asignaturas teóricas con alumnos en esta situación.

El uso de una base de datos colaboró a que los procesos fueron más rápidos, a que la información almacenada estuviera segura y de fácil acceso pero también fuera simple para los usuarios.

El sistema será un aporte por el momento dado su carácter de tesis para un sector de profesores del Departamento de Ingeniería en Computación y para el Departamento como tal permitiendo realizar las actividades relacionadas de manera más adecuada, pero en un futuro puede ser implementado en el resto de los departamentos, en la Facultad de Ingeniería en general o hasta la propia UNAM por completo.

El desarrollo no termina, es un proceso evolutivo, continuo e incluyente de las sugerencias de los usuarios para mejorarlo y de nuevos requisitos del cliente con el fin de conformar con el tiempo un sistema más robusto benéfico para todos.

Anexo

Scripts SQL para el traslado de información

Ejemplo de los scripts necesarios para copiar registros de tablas a otras con estructura diferente en Sybase:

```
select
ORGANO,NUMERO_UNICO,ASIGNATURA,ASIGNATURA_NOMBRE,NUMERO,IN
ICIO,FIN,CUPO,VACANTES,RFC,FOLIO,ACADEMICO_NOMBRE,LUN,MAR,MIE,
JUE,VIE,SAB,TIPO,SALON,NUMERO_PROFESOR,ASIGNATURA_AUX,ASIGNAT
URA_NOMBRE_AUX
into DIC_GRUPO
from  VISTA_GRUPO
go

/*********************************************

CREATE TABLE dbo.VISTA_GRUPO_BK
(
   ORGANO              int         NOT NULL,
   NUMERO_UNICO        int         NOT NULL,
   ASIGNATURA          smallint    NOT NULL,
   ASIGNATURA_NOMBRE   varchar(36) NOT NULL,
   NUMERO              smallint    NOT NULL,
   INICIO              smallint    NOT NULL,
   FIN                 smallint    NOT NULL,
   CUPO                smallint    NOT NULL,
   VACANTES            smallint    NOT NULL,
   RFC                 varchar(13) NOT NULL,
```

```
    FOLIO                int         NOT NULL,
    ACADEMICO_NOMBRE     varchar(60) NOT NULL,
    LUN                  varchar(1)  NOT NULL,
    MAR                  varchar(1)  NOT NULL,
    MIE                  varchar(1)  NOT NULL,
    JUE                  varchar(1)  NOT NULL,
    VIE                  varchar(1)  NOT NULL,
    SAB                  varchar(1)  NOT NULL,
    TIPO                 varchar(1)  NOT NULL,
    SALON                varchar(4)  NOT NULL,
    NUMERO_PROFESOR      varchar(1)  NOT NULL,
    ASIGNATURA_AUX       smallint    NULL,
    ASIGNATURA_NOMBRE_AUX varchar(36) NULL
)

select ASIGNATURA, ASIGNATURA_NOMBRE, TIPO
into DIC_ASIGNA
from VISTA_GRUPO
go

select
NUMERO_UNICO,ASIGNATURA,NUMERO,INICIO,FIN,FOLIO,LUN,MAR,MIE,JU
E,VIE,SAB,SALON
into DIC_GRUPO
from  VISTA_GRUPO
go

select RFC,FOLIO,ACADEMICO_NOMBRE
into DIC_PROF_COMP
from  VISTA_GRUPO
go
```

```
/************************************

CREATE TABLE dbo.INSCRITO

(
CUENTA       nvarchar(9) NULL,
NUMERO_UNICO int         NULL,
ASIGNATURA   int         NULL,
GRUPO        smallint    NULL
)


select A.CUENTA, B.NUMERO, A.ASIGNATURA,A.NUMERO_UNICO
into DIC_INSCRITO
from INSCRITO A, VISTA_GRUPO B
where A.ASIGNATURA=B.ASIGNATURA
and A.GRUPO=B.NUMERO
group by CUENTA,A.ASIGNATURA,A.GRUPO
go

select distinct A.CUENTA, B.NUMERO, A.ASIGNATURA,A.NUMERO_UNICO
into DIC_INSCRITO
from INSCRITO A, VISTA_GRUPO B
where A.ASIGNATURA=B.ASIGNATURA
and A.GRUPO=B.NUMERO
go

/************************************

CREATE TABLE dbo.ALUMNO
(
   CUENTA              varchar(9)  NOT NULL,
   NOMBRE              varchar(50) NOT NULL,
   PLN                 varchar(4)  NOT NULL,
   CARRERA             smallint    NOT NULL,
```

```
    PLN_DGAE            varchar(4)  NOT NULL,
    REGISTRO            varchar(5)  NOT NULL,
    SORTEO              int         NOT NULL,
    NIVEL1              smallint    NOT NULL,
    NIVEL2              smallint    NOT NULL,
    NIVEL3              smallint    NOT NULL,
    CREDITOS_CUBIERTOS  smallint    NOT NULL,
    CREDITOS_OBLIGATORIOS smallint  NOT NULL,
    CREDITOS_OPTATIVOS  smallint    NOT NULL,
    PROMEDIO            real        NOT NULL,
    AVANCE              real        NOT NULL,
    ESCOLARIDAD         real        NOT NULL,
    TEORIAS             smallint    NOT NULL,
    LABORATORIOS        smallint    NOT NULL,
    TIPO_MODULO         varchar(1)  NULL,
    ASIGNACION          smallint    NULL,
    LINEA               smallint    NOT NULL,
    CONTRASENA          varchar(5)  NULL
)

select CUENTA,NOMBRE,CARRERA
into DIC_ALUMNO
from ALUMNO
go


/*************************************

CREATE TABLE dbo.INSCRITO_EXTRAORDINARIOS
(
    CUENTA          varchar(9)  NOT NULL,
    NUMERO_UNICO    int         NOT NULL,
    ASIGNATURA      smallint    NOT NULL,
    GRUPO           smallint    NOT NULL,
    PERIODO         varchar(1)  NOT NULL,
    NOMBRE_ALUMNO   varchar(50) NULL,
    CARRERA         smallint    NULL,
    NOMBRE_CARRERA  varchar(50) NULL,
```

```
    NOMBRE_ASIGNATURA varchar(50) NULL,
    REGISTRO          varchar(5)  NULL
)

select  CUENTA, GRUPO NUMERO,
ASIGANTURA,PERIODO,REGISTRO,NUMERO_UNICO
into DIC_INSCRITO_EXTRA
from INSCRITO_EXTRAORDINARIOS
go
```

Bibliografía

- CEBALLOS, Fco. Javier. *JAVA 2: Interfaces gráficas y aplicaciones para Internet.* 3ª. Ed. Alfaomega Ra-Ma. México, 2008.
- CUENCA Jimenez, Pedro Manuel. *Programación en Java.* España: Anaya Multimedia, 1997.
- FRAMIÑÁN Torres, José Manuel. *Gestión de Bases de Datos en Internet: JDBC.* España: Anaya Multimedia, 1998.
- FROUFE Quintas, Agustín. *JAVA 2: Manual de usuario y tutorial.* 5ª. Ed. México: Alfaomega Ra-Ma, 2009.
- HALL, Marty. *Servlets y JavaServer pages: guía práctica.* México: Pearson Educación, 2001.
- LÓPEZ Román, Leobardo. *METODOLOGÍA DE LA Programación Orientada a Objetos.* México: Alfaomega, 2006.
- PRESSMAN, Roger. *Ingeniería del Software: Un enfoque práctico.* 5a. Ed. España: McGRAW-HILL, 2002.
- PRESSMAN, Roger. *INGENIERÍA DEL SOFTWARE: Un enfoque práctico.* 6a. Ed. México: McGraw-Hill Interamericana, 2005.
- RAMÍREZ, Felipe. *Introducción a la Programación.* 2ª.ed. México: Alfaomega, 2007.
- REESE, George. *Programación de bases de datos con JDBC y JAVA.* Madrid: Anaya Multimedia, 2001.
- TRIGOS García, Esteban. *JSP.* España: Anaya, 2001.
- SILBERSCHATZ, Abraham. *Sistemas Operativos. 6ª. Ed.* México: Limusa Wiley, 2004.
- STALLINGS, William. *Sistemas Operativos: Aspectos internos y principios de diseño.* 5ª. Ed. España: Pearson Educación, México: Prentice Hall, 2005.

Mesografía

- http://www.monografias.com/trabajos16/tecnologias-informacion/tecnologias-informacion.shtml
- http://pegaso.ls.fi.upm.es/~lmengual/ARQ_REDES/Arquitecturas_Seguridad.pdf
- http://mx.sun.com/practice/software/solaris/
- http://es.wikipedia.org/wiki/M%C3%A1quina_virtual_Java
- http://www.wikilearning.com/curso_gratis/gestion_de_riesgos_en_ingeniería_del_software-proyeccion_del_riesgo/3620-12
- http://eclases.tripod.com/id16.html
- http://alarcos.inf-cr.uclm.es/doc/pgsi/doc/teo/7/pgsi-t7.pdf
- http://www.monografias.com/trabajos41/riesgo-etapa-requisitos/riesgo-etapa-requisitos2.shtml
- http://www.mitecnologico.com/Main/FormulacionDeLosEstudiosDeFactibilidadTecnicaEconomicaYOperativaDelProyecto
- http://boards4.melodysoft.com/2005AISI0406/-el-plan-de-software-estudio-de-factibilidad-15.html
- http://www.cid.uc.edu.ve/fponte/ejemplo/factib.pdf
- http://www.eumed.net/ce/2009a/amr.htm
- http://www.v880.net/
- http://download.oracle.com/docs/cd/E19095-01/sfv880.srvr/index.html

- http://en.wikipedia.org/wiki/Sun_Fire
- http://sistemas.itlp.edu.mx/tutoriales/basedat1/tema1_9.htm
- http://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_bases_de_datos
- http://es.wikipedia.org/wiki/Adaptive_Server_Enterprise
- http://www.sybase.es/products/databasemanagement/adaptiveserverenterprise
- http://dis.um.es/~jnicolas/09BK_FIS.html
- http://www.cyta.com.ar/biblioteca/bddoc/bdlibros/proyectoinformatico/libro/c4/c4.htm
- http://profesores.fi-b.unam.mx/heriolg/Apa_1apa.pdf
- http://es.wikipedia.org/wiki/Diagrama_de_Flujo_de_Datos
- http://documentos.mideplan.go.cr/alfresco/d/d/workspace/SpacesStore/6a88ebe4-da9f-4b6a-b366-425dd6371a97/guia-elaboracion-diagramas-flujo-2009
- .pdf
- http://es.wikipedia.org/wiki/JQuery
- http://sarfraznawaz.wordpress.com/2010/09/19/styletable-jquery-plugin/
- http://tooltipsy.com/
- http://www.styleshout.com/
- http://www.itextpdf.com/

- http://librojquery.com/
- http://www.itextpdf.com/book/examples.php
- http://www.it.uc3m.es/labttlat/lab5/
- http://bulma.net/body.phtml?nIdNoticia=1888
- http://neossoftware.260mb.com/2009/06/configuracion-de-tomcat-55-en-eclipse/
- http://mundogeek.net/archivos/2009/02/02/eclipse-y-tomcat/