



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ESTA TESIS NO SALI
DE LA BIBLIOTECA

MODELO DESCRIPTIVO A TRAVÉS DEL USO DE
HERRAMIENTAS DE INGENIERÍA DE SOFTWARE
EMPLEADO EN LA COORDINACIÓN DE
LABORATORIOS DE INGENIERÍA MECÁNICA.

TRABAJO ESCRITO

Que para obtener el título de:

Ingeniera en Computación.

PRESENTAN

Miriam Graciela Mendoza Cano

Yenni Quintana Sánchez

Director:

Ing. Jesús Roviroza López.

Codirector:

Ing. José Arturo Origel Coutiño.





Universidad Nacional
Autónoma de México

Dirección General de Bibliotecas de la UNAM

Biblioteca Central



UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

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

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

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ESTA TESIS NO SALE
DE LA BIBLIOTECA



LABORATORIOS DE INGENIERÍA MECÁNICA.
EMPLEADO EN LA COORDINACIÓN DE
HERRAMIENTAS DE INGENIERÍA DE SOFTWARE
MODELO DESCRIPTIVO A TRAVÉS DEL USO DE

TRABAJO ESCRITO

Que para obtener el título de:

Ingeniería en Computación.

PRESENTA

Miriam Graciela Mendoza Cano

Yenni Quintana Sánchez

Director:

Ing. Jesús Rovirosa López

Codirector:

Ing. José Arturo Origel Cordero



Agradecimientos

Miriam Graciela Mendoza Cano

A mi mamá Graciela Cano Carrillo

Por enseñarme a ser la persona que soy e impulsarme en todos los momentos difíciles y buenos de mi vida sin importar las adversidades así como la infinita paciencia que tiene conmigo.

A mi hermano José Gabriel Mendoza Cano

Por ser el más pequeño de edad, pero el más grande en fuerza y voluntad; aspirando y logrando siempre un ser triunfador que me inyecta día a día la actitud y entereza para lograr mis metas. Muchas gracias.

A mi hermana Irma Adriana Mendoza Cano

*Por contagiarme de su fortaleza de carácter y reincidir una y otra vez hasta lograr los objetivos trazados en mi vida; y a mi sobrina **María Fernanda González Mendoza** por su sana y alegre inocencia, viendo siempre al futuro, dan un toque de alegría.*

A mi papá Gabriel Mendoza de la Vega

Porque con el ejemplo de sus logros en la vida motivo mis deseos de superación y progreso para mi propio crecimiento profesional. Gracias papá Gabriel.

A Edith Mijares Arellano y Eduardo Compañ Órnelas

Por ser mis mejores amigos a lo largo de muchos años. Hemos compartido muchas cosas juntos y les agradezco infinitamente todo lo que me han brindado este tiempo.

A Estela Flores-Magón Jiménez.

Porque es una de las personas que he conocido con mucha paciencia. La tranquilidad de tu forma de ser me ha vuelto a enseñar el valor de las cosas más simples de la vida.

A la DIMEI.

La oportunidad de trabajar con el Ing. Moisés Mendoza Linares, el Ing. Héctor Mejía Ramírez, el Ing. Víctor M. Vázquez Huarota, la Lic. Ma. Teresa Yebra García, el señor Joaquín Trejo Ramírez; la Lic. Luz del Carmen Sosa Hernández y las secretarías María del Carmen Fernández Viano, Lilia Cerón Vaquero y Cecilia Correa; así como el jefe de la División, el M. en C. Enrique Jiménez Espriú ha sido de mucha ayuda en mi vida ya que he aprendido muchas cosas de cada uno de ellos tanto académica como profesionalmente.

A Yenni Quintana Sánchez.

El conocerte personal y académicamente a lo largo de la carrera, ha sido y serán uno los momentos más gratos de mi vida. Cuando decidimos emprender este proyecto, nunca pensé en todas las cosas buenas y malas que hemos pasado para llevarlo a cabo. Pero sin duda, nos han acercado a ser mejores amigas. Mil gracias por ello.

A la Facultad de Ingeniería y a la Universidad Nacional Autónoma de México.

El estudiar en el mejor espacio académico y cultural para adquirir el conocimiento de Ingeniería en Computación lo he logrado gracias a la Facultad de Ingeniería de la Universidad Nacional Autónoma de México. Muchas gracias por permitirme lograr mi formación profesional en ésta institución.

Yenni Quintana Sánchez

A mi Madre Concepción Sánchez Rodríguez

Por hacer de mí la persona que soy y darme lo mejor de su vida, por dedicarse a darme todo cuanto esta en sus manos, por enseñarme las cosas agradables en los problemas que puedan presentarse y por ser mi Madre.

A mis Tios: Celia Sánchez Rodríguez y Guillermo López Armijo.

Por cobijarme en su familia, por tenerme como hija y por todo el amor, dedicación y cuidados que siempre me han dado, por ser mis segundos padres.

A mis Primos: Javier, Verónica y Claudia López Sánchez.

Por ser mis hermanos y darme un ejemplo de unión, trabajo y dedicación, por todos esos años compartidos.

A mi Familia

Por los momentos compartidos en los mejores y peores momentos, a mi abuelo Ángel Sánchez Uribe por ese corto tiempo vivido.

A Yuri

Por estar presente en los momentos importantes, por darme lo mejor de su persona y por el camino andado juntos.

A todos mis Amigos

Por ser parte importante en mi vida y carrera, por los días compartidos, la paciencia y apoyo brindado. A los amigos de la infancia: Julio Alberto Meza Ruíz, a todos los amigos de la Universidad que me motivaron a concluir este trabajo.

Al Centro de Docencia

Al Ing. Carlos Sánchez Mejía, por ser un gran impulso personal y profesionalmente, por la confianza depositada en mí y la amistad brindada, por permitir mi desarrollo profesional; a todos los amigos que allí se encuentran: Lic. Claudia, Lic. Martha Rosa, Martha Elena, Lic. Mar, Alejandro, Eduardo y Ana Bautista.

A la Universidad Nacional Autónoma de México.

Especialmente a la Facultad de Ingeniería por permitirme obtener una educación personal y académica que me permitirá desarrollarme en cualquier ámbito profesional y ser una persona de calidad. Por ser mi Alma Máter.

A la DIMEI

Al Ing. Moisés Mendoza Linares, el Ing. Héctor Mejía Ramírez, el Ing. Víctor M. Vázquez Huarota por permitirme realizar el servicio social y brindarme su amistad, la Lic. Ma. Teresa Yebra García todo el apoyo y orientación que me brinda, el señor Joaquín Trejo Ramírez y las secretarias María del Carmen Fernández Viano, Lilia Cerón Vaquero; así como el jefe de la División, el M. en C. Enrique Jiménez Espriú.

A Miriam

Por compartir parte de su vida, por aguantar las crisis sufridas y por la amistad que me brinda.

Y queremos darles un agradecimiento **muy especial.**

A la Ing. Norma Ruth Leonar Pérez

Por ser nuestra mejor amiga durante toda la carrera. Siempre nos has motivado y nos has ayudado en todo momento. También te damos las gracias por tus consejos y regaños que siempre nos enseñan las cosas claras de las situaciones difíciles y buenas de la vida.

Al Lic. Omar Ávila del Valle

Por enseñarnos la bondad que un profesionalista debe tener en el uso del conocimiento, por dedicarnos parte de sus noches en la enseñanza de lenguajes de programación, la enorme paciencia con nosotras y brindarnos su amistad desinteresadamente.

Y a nuestros amigos y maestros que nos ayudaron, en todos los sentidos, durante la carrera y a la realización de este trabajo.

***M. en I. Silvina Hernández García.
Lic. Miguel Angel Mejía Argueta.
Ing. Armando Sánchez Guzmán.
Ing. Laura Sandoval Montaña.
José Elesban López Ovando.
Ing. Adolfo Millán Nájera.
Ana Claudia Reyes Cruz.
Delfino Saavedra Rivera
Ing. Noé Cruz Marín.
Beatriz Tinajero.***

Sin olvidar a nuestros directores del trabajo

Ing. Jesús Roviroza López e Ing. José Arturo Origel Coutiño

¡Muchas Gracias!

ÍNDICE

<u>PRÓLOGO.</u>	i
<u>INTRODUCCIÓN.</u>	iii
<u>CAPÍTULO 1. "PLANTEAMIENTO DEL PROBLEMA"</u>	1
Etapas de Inscripción.	1
Diagrama de flujo de la información del proceso de inscripción de la Facultad de Ingeniería.	2
Diagrama de flujo de la información de cambios de grupo de teoría y movimientos de laboratorio.	3
Diagrama de flujo de la información utilizada por el departamento de Ingeniería Mecánica durante el proceso de inscripción a los laboratorios (proceso interno).	3
Diagrama de contexto para la inscripción en la facultad de ingeniería.	4
Diagrama de flujo de datos para la inscripción en la facultad de ingeniería.	4
Diagrama de flujo del proceso 1.	5
Diagrama de flujo del proceso 2.	5
Diagrama de flujo del proceso 3	6
Diagrama de flujo del proceso 2.1	6
Diagrama de flujo actual del proceso 2.1.6	7
Objetivo e hipótesis	7
<u>CAPÍTULO 2. "SELECCIÓN DE LA METODOLOGÍA PARA EL DESARROLLO DE LA PROPUESTA".</u>	9
Modelo Lineal Secuencial.	10
Modelo de Construcción de Prototipos.	13
Modelo DRA.	14
Modelo Incremental.	16
Modelo en Espiral.	17
Modelo de Ensamblaje de Componentes.	18
Modelo de Desarrollo Concurrente.	20
Modelo de Métodos Formales.	21

Diagramas de contexto y flujo de datos	21
Representación gráfica de los diagramas de flujo de datos	22
Selección del método a utilizar.	23
<u>CAPÍTULO 3. "MANEJO DE BASES DE DATOS"</u>	25
Base de datos	25
Objetivos de una base de datos.	25
Costos	25
Ventajas de las bases de datos.	25
Independencia lógica de los datos.	26
Independencia física de los datos	26
Bases de datos relacionales (o modelo de datos relacional).	27
Reglas de Codd	27
Sistema manejador de bases de datos	28
Estructura General de un DBMS.	29
Componentes del Sublenguaje Empleado por el DBMS.	29
Lenguaje de Definición de Datos, LDD (Data Definition Language, DDL).	29
Lenguaje de Manipulación de Datos, LMD (Data Manipulation Language, DML).	29
Diccionario de Datos, DD (Data Dictionary, DD).	30
Lenguaje de Control de Datos, LCD (Data Control Language, DCL)	31
El enfoque relacional.	31
Diseño de bases de datos.	31
Modelo Conceptual	31
Modelo Lógico	31
Modelo Físico	32
Diagrama entidad-relación	32
Normalización	34
Primera forma normal	34
Segunda forma normal	34
Tercera forma normal	34

<u>CAPÍTULO 4. "SELECCIÓN DEL LENGUAJE DE PROGRAMACIÓN"</u>	37
Visual Basic.	37
Java.	38
PHP.	39
Power Builder.	40
Elección del lenguaje de programación.	47
<u>CAPÍTULO 5. "REDES DE COMPUTADORAS."</u>	43
Redes de computadoras	43
Ventajas de las Redes de Computadoras.	43
Clasificación de las Redes de Computadoras.	43
a) De acuerdo a su alcance geográfico	43
b) De acuerdo a su topología	43
Transmisión de datos en redes	48
Protocolos	48
OSI y TCP/IP	49
Seguridad en las Redes de Computadoras.	51
Firewalls	51
Redes Privadas Virtuales.	51
Antivirus	52
Seguridad de e-mail/contenido web	52
Integridad del negocio	52
Integridad de la red	52
Autenticación de dos factores	53
Detección de intrusos	53
Https	54
Arquitectura cliente/servidor	54
Características de un sistema cliente/servidor	54
Partes de un sistema cliente/servidor	55
La sección frontal	55
La sección posterior	55
Tipos de servidores	56
1. Servidores iterativos	56

2. Servidores concurrentes	56
Arquitectura de 2 capas	57
Arquitectura de 3 capas	57
Ventajas e inconvenientes de la arquitectura cliente/servidor	58
Análisis de algunos sistemas gestores de bases de datos (DBMS) comerciales	59
Postgresql	59
Oracle	60
Access	62
Power Builder	63
Análisis de la red actual en USECAD	63
<u>CAPÍTULO 6. "PRESENTACIÓN DE LA PROPUESTA".</u>	65
Diagrama de flujo propuesto	65
Diagrama de flujo propuesto de la información de movimientos de inscripción de laboratorios en el departamento de Ingeniería Mecánica. (Proceso interno)	65
Tablas de USECAD a consultar	66
Modelo Entidad Relación de la Propuesta	67
Requerimientos de Hardware.	68
Interfase de la propuesta en Power Builder	68
<u>CAPÍTULO 7. "DESCRIPCIÓN DE PRUEBAS Y POSIBLES RESULTADOS".</u>	71
Aspectos de la instrumentación de la Programación y Programación Modular.	71
Pruebas	71
El proceso de Prueba	72
Modularidad en Pruebas.	72
Pruebas de Unidad	74
Pruebas del Sistema	75
Pruebas de Validación	76
Pruebas Alfa	76
Pruebas Beta	77
Desarrollo de Pruebas a la aplicación para la inscripción a los laboratorios del departamento de Ingeniería Mecánica	77
Alcance de este Trabajo	78
Resultados esperados	79

CONCLUSIONES 81

GLOSARIO 83

BIBLIOGRAFÍA. 93

PRÓLOGO

Este trabajo escrito es una propuesta a través del uso de herramientas de software, para permitir establecer una comunicación entre el Departamento de Ingeniería Mecánica de la DIMEI (División de Ingeniería Mecánica e Industrial) y la USECAD (Unidad de Servicios de Cómputo Administrativos); y llevar a cabo el proceso de inscripción a los laboratorios de las asignaturas que cursan los alumnos de las carreras de Ingeniería Mecánica e Ingeniería Industrial.

En la **Introducción** se presenta un resumen, muy breve, del avance tecnológico de la humanidad. Así como también una descripción del problema al inscribir a los alumnos a los grupos de laboratorio en el Departamento de Ingeniería Mecánica.

En el capítulo uno, **planteamiento del problema**, se describe el proceso que efectúa el Departamento de Ingeniería Mecánica y los alumnos para realizar las inscripciones a los laboratorios dentro del proceso general que cumple cada semestre la Facultad de Ingeniería; la problemática del departamento para realizar el proceso; así como el objetivo y la hipótesis de la propuesta que tenemos para solucionar ésta problemática.

En el capítulo dos, **selección de la metodología para el desarrollo de la propuesta**, a través de los conceptos de ingeniería de software, planteamos una metodología (serie de pasos a seguir) para realizar nuestra propuesta; desde solicitar y analizar los requerimientos del usuario hasta el planteamiento de un programa hecho con herramientas de software.

En el capítulo tres, **manejo de bases de datos**, se describen algunos conceptos de éste tema ya que parte de la propuesta, es la creación de una base de datos que permita al usuario administrar la información para realizar las inscripciones en el Departamento de Ingeniería Mecánica.

En el capítulo cuatro, **selección del lenguaje de programación**, se describen las características de algunos lenguajes de programación que nos permiten construir interfaces gráficas para acceder a las bases de datos.

En el capítulo cinco, se presenta el análisis de **redes de computadoras**, se hace un resumen de la arquitectura de las mismas así como la descripción del funcionamiento de la red que nos permitirá establecer una comunicación con la USECAD.

En el seis, **presentación de la propuesta**, se hace una representación gráfica del planteamiento de la solución; empezando por el uso del diagrama de procesos, el modelo conceptual a través del diagrama entidad/relación; hasta la muestra del despliegue de ventanas realizadas en PowerBuilder que representan el funcionamiento de la base de datos de nuestra propuesta.

En el capítulo siete, **descripción de pruebas y posibles resultados**, se hace un planteamiento de una secuencia de pruebas a realizar en la propuesta del programa para verificar y darle mantenimiento para su funcionamiento óptimo.

Finalmente, se agrega un apartado para las **conclusiones** y un **glosario** de términos empleados a lo largo de este trabajo escrito.

INTRODUCCIÓN

Hoy nos podemos preguntar si la aventura del hombre en su desarrollo llegará a su propia conquista biológica, de la técnica y de la inteligencia.

En su desarrollo, la etapa industrial ha conquistado, gracias a la energía, a las grandes máquinas, pero éstas requieren gran parte de ella que nos es aventurado pensar en su desaparición como consecuencia de una crisis energética, y con ella la de la sociedad. Mientras tanto, el área informática prolifera y vislumbra perspectivas grandiosas, contemplando en la actualidad cambios inimaginables. Estas perspectivas son el relevo de la era de la informática, con un futuro promisorio.

En la era de la teleinformática, los hombres: pueden comunicarse a distancia e informarse del contenido de muchas fuentes de información y del resultado de sus investigaciones.

Aún con tantos adelantos, resulta difícil de calcular lo que significa este recurso: podemos consultar, desde nuestra casa u oficina, cualquier banco de datos para conocer desde los horarios de los aviones hasta los medicamentos y sus efectos. La revolución de la informática anuncia, sin duda, un inminente cambio social.

La historia de la electrónica es vertiginosa, los adelantos y sus aplicaciones se suceden sin descanso, los precios de sus componentes bajan y los artefactos se encuentran al alcance de muchos. Hoy encontramos computadoras con el software específico para toda clase de trabajos, así nuestro cerebro puede confiar en ellas, encomendándoles que realicen los cálculos y almacenen los datos, mientras desarrollamos nuestra creatividad.

Con el avance tecnológico también llegó el avance en el manejo de información, lo que podemos observar con el surgimiento de la tecnología de las bases de datos la cual es una de las áreas de la ciencia de la computación y la información de más rápido desarrollo. Los fabricantes y vendedores no empezaron a ofrecer sistemas de administración de bases de datos hasta mediados de la década de 1960 (aunque es verdad que ciertos paquetes de software antiguos incluían algunas de las funciones que ahora se asocian con tales sistemas). Pese a su calidad de innovación, sin embargo, el campo rápidamente ha cobrado importancia práctica y teórica.

En el núcleo de muchas aplicaciones empresariales de software reside una base de datos. Las bases de datos prevalecen en el mundo empresarial debido a que permiten el acceso centralizado a la información de una manera uniforme y eficaz, y relativamente fácil de configurar y mantener.

Es por esto que dentro de la Facultad de Ingeniería, institución educativa con una gran tradición en la formación de Ingenieros en sus doce ramas del conocimiento, nos avocamos a llevar adelante esa actualización, que grano a grano del conocimiento se va extendiendo cada vez más.

Es así, que dentro de toda la red de movimientos administrativos requeridos para la admisión, registro y control de sus alumnos, maneja el proceso de inscripción de los mismos, cada semestre, a las diferentes opciones en la toma de asignaturas, éstas con su parte práctica (laboratorios), para llegar a cubrir los créditos necesarios en cada especialidad.

Dicho proceso, es el resultado de una continua búsqueda de actualización del mismo y en particular lo referente a las inscripciones de la parte práctica de las asignaturas, sus laboratorios; con la finalidad

de realizar de manera más eficiente el proceso, y que contemple características como: compaginación de los diferentes lenguajes y manejo de información involucrada, con una larga vida útil y optimización del proceso de inscripción a los laboratorios en general.

Dentro de la Facultad de Ingeniería, en su proceso general de inscripción a las asignaturas solicitadas por los alumnos semestre a semestre, se realiza un proceso paralelo que es la inscripción complementaria a los laboratorios de las materias que curricularmente se cursan en el departamento de Ingeniería Mecánica.

Dado que el proceso de inscripción complementaria a los laboratorios del departamento de Ingeniería Mecánica se realiza de forma manual, lenta, tediosa y con problemáticas continuas, surge la necesidad de diseñar una propuesta para que el proceso se lleve a cabo de manera directa, y segura; ahorrando duplicidad de trabajos y redundancia o pérdida de la información, así como agilizar el registro de los movimientos realizados durante el proceso de inscripciones complementarias y realizar el almacenamiento de calificaciones obtenidas en semestres anteriores para que el alumno pueda revalidar la acreditación de algún laboratorio que le permita cursar únicamente la teoría.

Este trabajo está aplicado únicamente a las inscripciones complementarias de los laboratorios de Ingeniería Mecánica de la Facultad de Ingeniería, ya que su utilidad y funcionamiento, permitirá al departamento administrar su información para el manejo interno de altas, bajas, cambios, generación de consultas y listados de acuerdo a sus necesidades.

CAPÍTULO 1

PLANTEAMIENTO DEL PROBLEMA

El proceso de inscripciones a los laboratorios del Departamento de Ingeniería Mecánica, así como en otros departamentos de las diferentes Divisiones de la Facultad de Ingeniería, no ha logrado actualizarse a través del tiempo como es el caso; hoy en día, de las asignaturas correspondientes a la parte teórica. Dicho proceso se realiza regularmente en tres y hasta cuatro etapas, que debe realizar el alumno para poder obtener finalmente el horario, grupo y profesor de laboratorio deseado para que compagine con todas sus asignaturas y otros laboratorios que debe cursar durante un semestre y así cubrir los requerimientos de los niveles designados en el plan de estudios de su carrera.

Para poder tener una inscripción definitiva, el alumno debe pasar, por lo menos, por dos de las siguientes **etapas de inscripción**:

1. **Inscripción regular** donde el alumno tiene el derecho a realizar de forma personal su inscripción a las asignaturas de teoría y sus correspondientes grupos de laboratorio.

Ventajas:

- El alumno con bajo número de inscripción, sin problema puede acceder a que se le asignen todos los grupos de teoría y laboratorios solicitados por él, de acuerdo a sus requerimientos. Por lo tanto, pasaría directamente a la cuarta etapa de inscripciones, para nuestro caso.

Desventajas:

- De no obtener todas las asignaturas deseadas, ya sea de teoría o de laboratorio, el alumno tiene que acceder a la segunda etapa de inscripción.
- Puede obtener sus grupos de teoría y no los solicitados para sus laboratorios.

2. **Inscripciones de casos especiales¹ (Altas y bajas).**

Ventajas:

- El alumno logra obtener todas las asignaturas con todos sus laboratorios en los horarios deseados.

Desventajas:

- En caso de no obtener todos los grupos de laboratorio, tiene que acceder a la tercera etapa de inscripción.

¹ Estos se dan cuando por alguna circunstancia el alumno no se inscribió en fecha y forma correspondiente.

3. Cambios de grupo de teoría y Altas, Cambios y Bajas de Laboratorios.

Ventajas:

- El alumno logra obtener todas sus asignaturas y laboratorios en los horarios deseados a través del proceso de cambio de grupos, en el cual es posible generar altas, cambios y bajas de grupo de laboratorios de manera interna en el Departamento de Ingeniería Mecánica.

4. **Confirmación de inscripción y generación de credenciales.** Aunque ya tenga designados todos los grupos de laboratorio, el alumno deberá pasar, en el caso de Asignaturas del Departamento de Mecánica, a un proceso final de confirmación de inscripción de sus laboratorios, con la finalidad de que se revise el cupo final del grupo, o bien, otros procedimientos adicionales según sea el caso del laboratorio como el entregar una fotografía para una credencial interna de manejo de equipo en el laboratorio correspondiente.

Con el siguiente diagrama explicaremos la forma en la cual se lleva a cabo el proceso de reinscripciones en la Facultad de Ingeniería, con ello comprenderemos cual es el flujo de la información entre el Departamento de Ingeniería Mecánica y USECAD; así mismo demostraremos la necesidad de crear la propuesta que permita optimizar recursos tiempo y reducir pérdida de información.

NOTA: La explicación del significado y uso de los siguientes diagramas se realiza en el Capítulo 2.

DIAGRAMA DE FLUJO DE LA INFORMACIÓN DEL PROCESO DE INSCRIPCIÓN DE LA FACULTAD DE INGENIERÍA

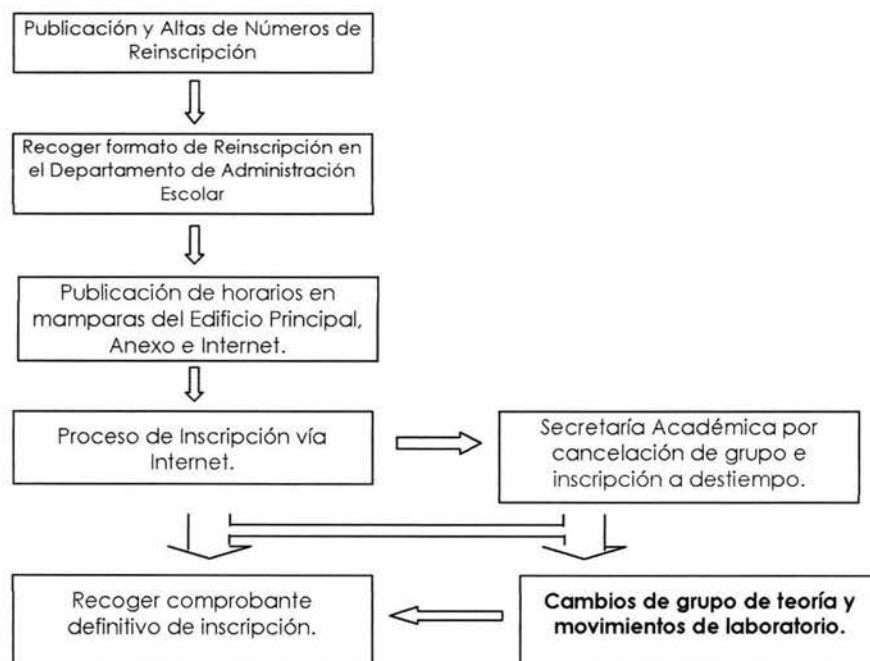


DIAGRAMA DE FLUJO DE LA INFORMACIÓN DE CAMBIOS DE GRUPO DE TEORÍA Y MOVIMIENTOS DE LABORATORIO.

Este diagrama representa el flujo de información durante los cambios de grupo en la inscripción ordinaria en la Facultad de Ingeniería.

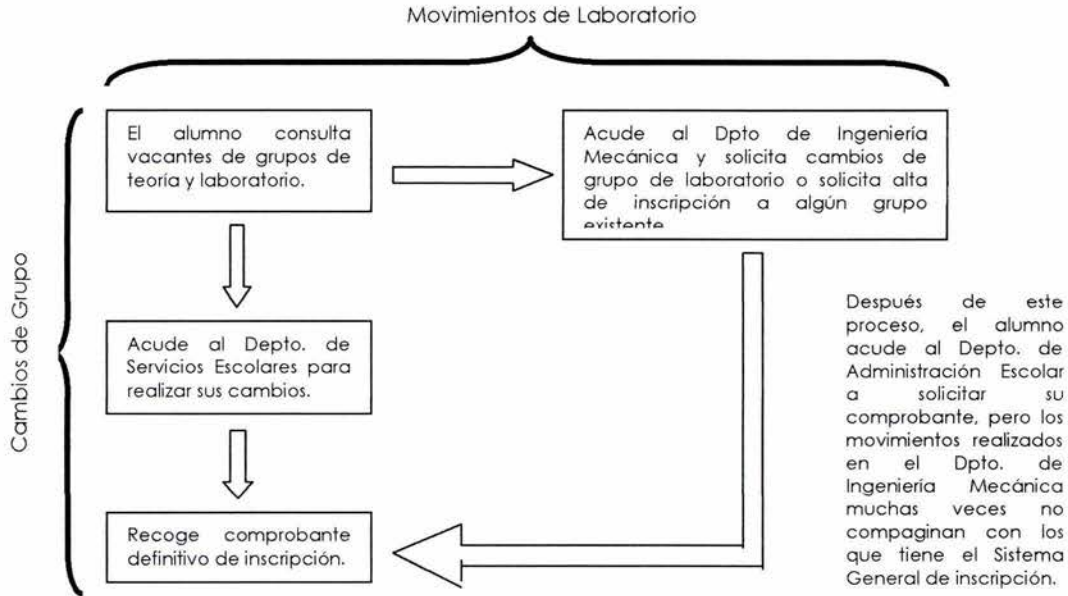


DIAGRAMA DE FLUJO DE LA INFORMACIÓN UTILIZADA POR EL DEPARTAMENTO DE INGENIERÍA MECÁNICA DURANTE EL PROCESO DE INSCRIPCIÓN A LOS LABORATORIOS (PROCESO INTERNO).

Este diagrama representa el flujo de información durante el periodo de inscripciones en el Departamento de Ingeniería Mecánica.

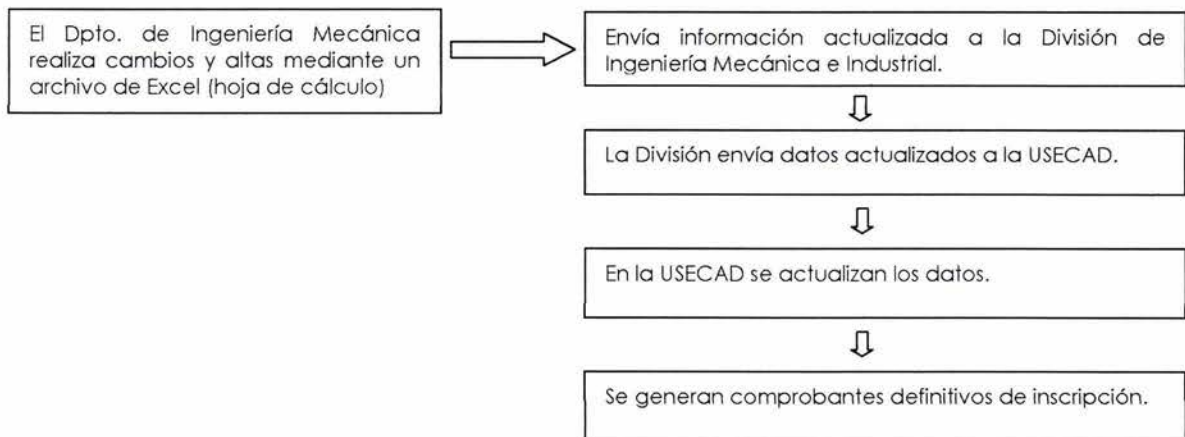


DIAGRAMA DE CONTEXTO PARA LA INSCRIPCIÓN EN LA FACULTAD DE INGENIERÍA.

Los siguientes diagramas muestran cada uno de los subprocesos que conforman el proceso de inscripciones. Se inicia con el proceso ordinario hasta finalizar con el que realiza de manera interna el Departamento de Ingeniería Mecánica.



DIAGRAMA DE FLUJO DE DATOS PARA LA INSCRIPCIÓN EN LA FACULTAD DE INGENIERÍA.

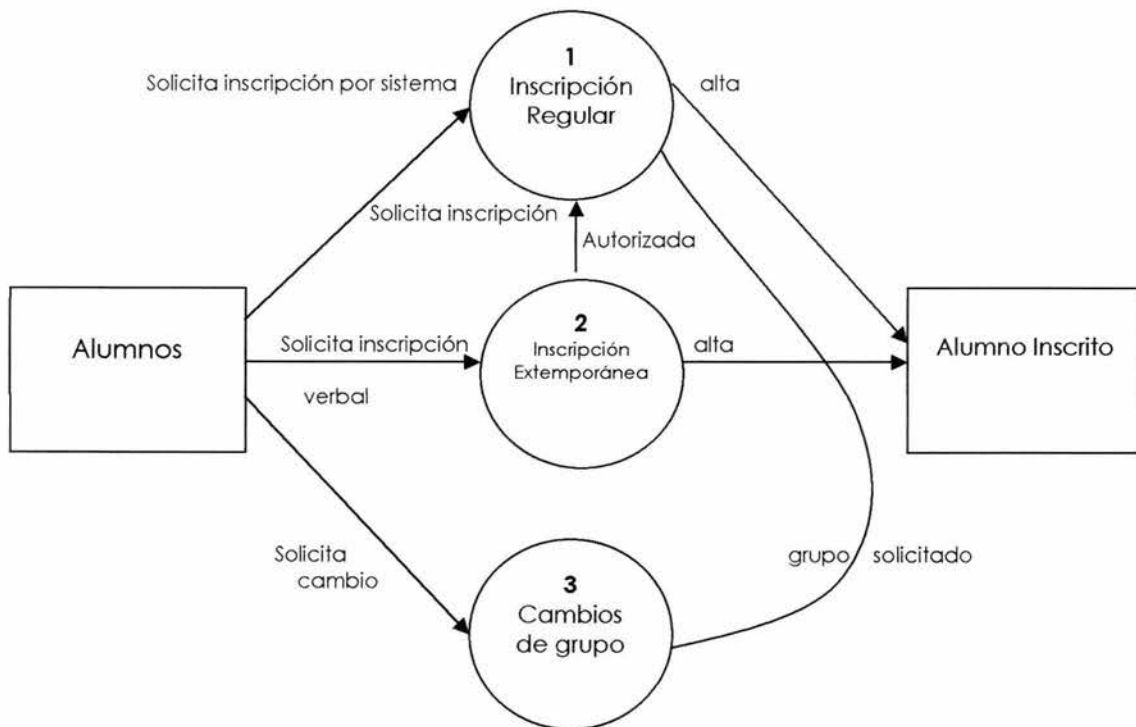


DIAGRAMA DE FLUJO DEL PROCESO 1

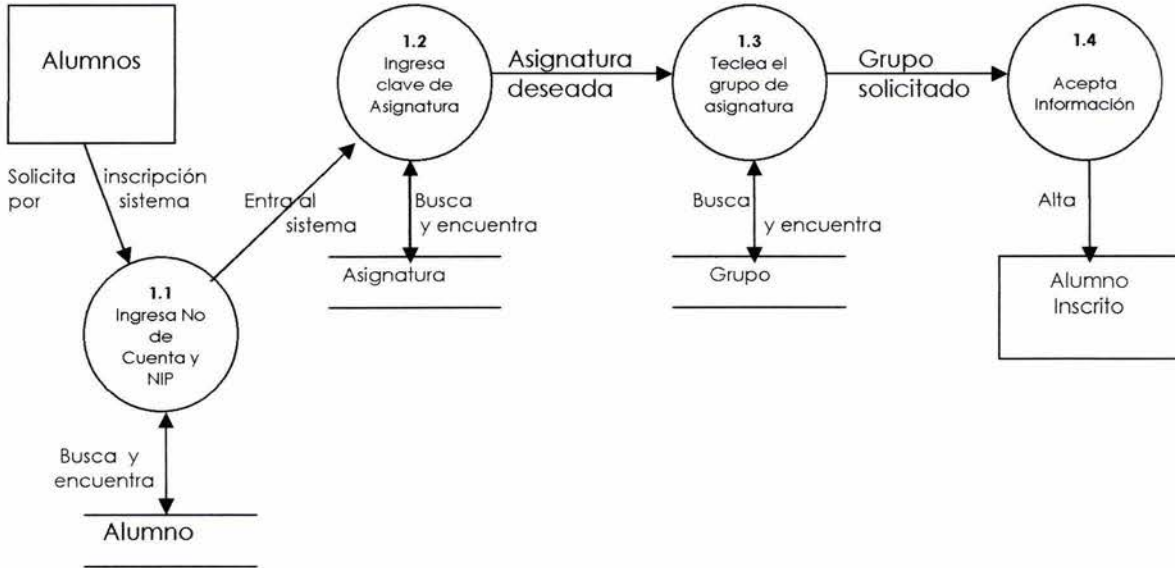
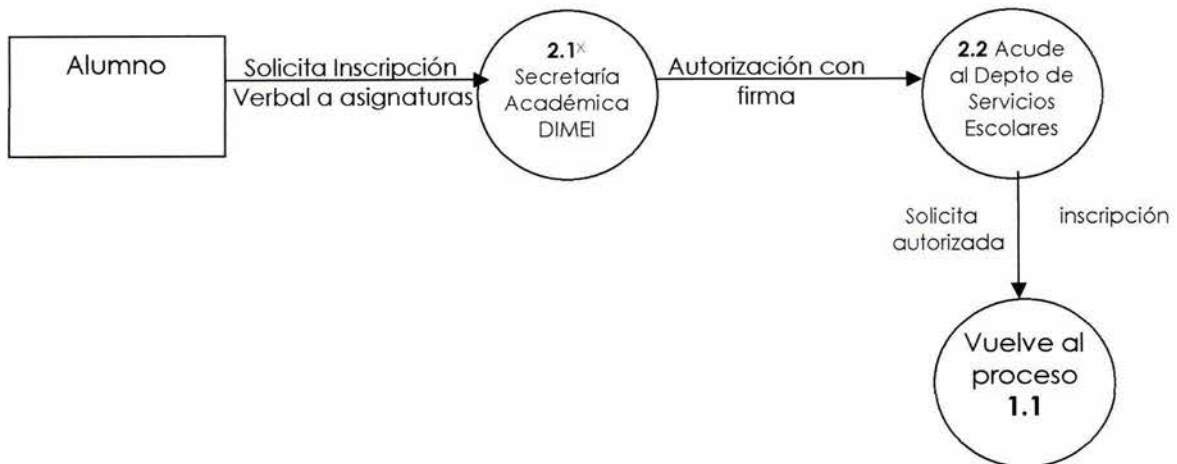


DIAGRAMA DE FLUJO DEL PROCESO 2



× A partir de este proceso, la descripción se refiere a la inscripción complementaria realizada en el Departamento de Ingeniería Mecánica actualmente.

DIAGRAMA DE FLUJO DEL PROCESO 3

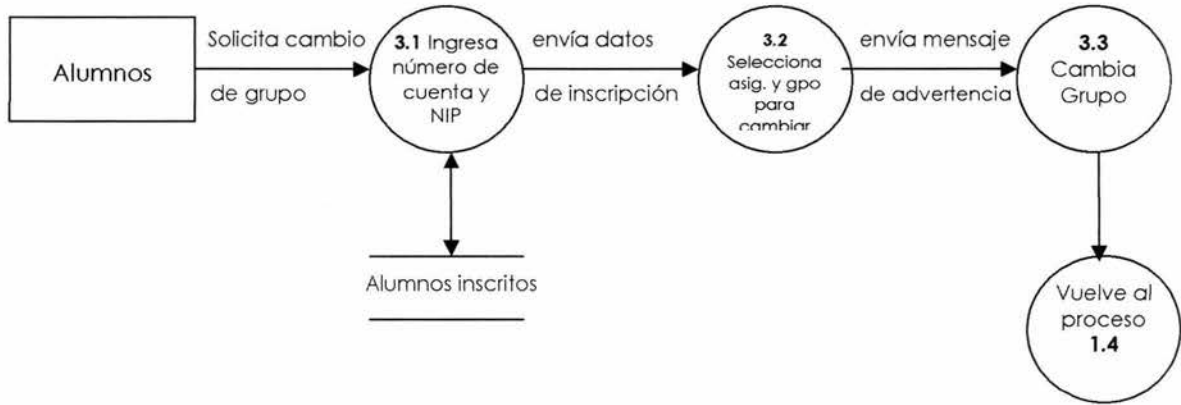


DIAGRAMA DE FLUJO DEL PROCESO 2.1

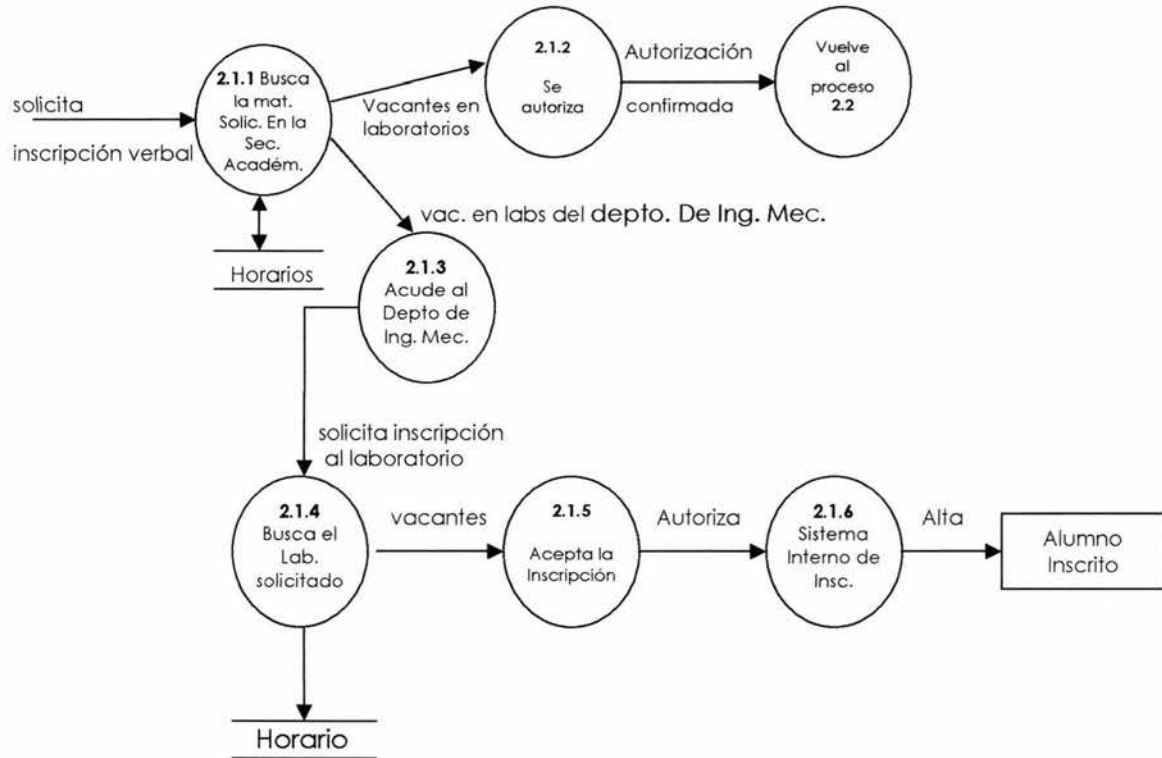


DIAGRAMA DE FLUJO ACTUAL DEL PROCESO 2.1.6



Es así como el proceso de inscripciones a los laboratorios de Ingeniería Mecánica se lleva a cabo de una forma lenta, tediosa e insegura. Se tiene además un gran retraso en la generación de actas tanto provisionales como definitivas, así mismo en los listados de asistencia, calificaciones y credenciales para el control de herramienta en cada uno de los laboratorios.

Debido a todos los motivos anteriormente citados, surge la necesidad de crear una propuesta que permita obtener esta información desde la base de datos de USECAD, esta información debe obtenerse de manera oportuna lo cual implica trabajar con un lenguaje de programación que sea lo suficientemente robusto como para garantizar la seguridad de la información durante su envío.

Por lo anteriormente mencionado tenemos como **objetivo** realizar una propuesta que permita automatizar el proceso de inscripción complementaria de alumnos a los laboratorios correspondientes al departamento de Ingeniería Mecánica; contemplando altas, bajas, cambios de grupo, asignación de profesores, listados preliminares, definitivos y elaboración de credenciales para el acceso, control y manejo de herramientas en cada laboratorio; lo cual nos lleva a establecer como **hipótesis** de este trabajo lo siguiente:

- Al concluir este trabajo, se ofrecerá una propuesta que le permita al usuario final llevar a cabo los movimientos que los alumnos requieran para finalizar su inscripción semestral; así mismo, realizar las impresiones de listados y credenciales requeridos.
- De igual forma, el usuario final podrá mantener un registro del semestre anterior de los alumnos que hayan acreditado el laboratorio, no así la teoría para que puedan revalidar dicha calificación.
- La puesta en marcha de la aplicación generada con nuestra propuesta resultaría conveniente pues ahorraría el tiempo de inscripción y proporcionaría al Departamento de Ingeniería Mecánica un mejor servicio de atención a los alumnos.

CAPÍTULO 2

SELECCIÓN DE LA METODOLOGÍA PARA EL DESARROLLO DE LA PROPUESTA

Para el desarrollo de este trabajo, se analizarán diferentes procesos de la Ingeniería de software, con el objeto de identificar los pasos que deben seguirse para llevar a cabo el diseño de nuestra propuesta, desde el conocimiento del problema hasta la puesta en marcha y realización de mantenimiento de la misma.

Entre los procesos de software encontrados están los siguientes:

- a. Modelo Lineal Secuencial.
- b. Modelo de Construcción de Prototipos.
- c. Modelo de Desarrollo Rápido de Software (DRA).
- d. Modelo Incremental (extensión del Modelo Lineal Secuencial).
- e. Modelo en Espiral (extensión del Modelo de Construcción de Prototipos).
- f. Modelo de Ensamblaje de Componentes.
- g. Modelo de Métodos Formales.

Todo el desarrollo del software se puede caracterizar como un ciclo de resolución de problemas en el que se encuentran cuatro etapas distintas, las cuales son¹:

1. *Status quo*. Representa el estado actual de sucesos.
2. Definición de problemas. Identifica el problema específico a resolverse.
3. Desarrollo técnico. Resuelve el problema a través de la aplicación de alguna tecnología.
4. Integración de soluciones. Ofrece los resultados a los que solicitan la solución en primer lugar.

El ciclo de resolución de problemas se aplica al trabajo de ingeniería del software en muchos niveles diferentes de resolución. Se puede utilizar en el macro nivel cuando se tiene en consideración la aplicación entera; en un nivel medio cuando se está considerando los componentes del programa, e incluso en la línea del nivel de código. Por consiguiente, se puede utilizar una representación fractal para proporcionar una visión idealizada del proceso. En la figura 1, cada etapa del ciclo de resolución de problemas contiene un ciclo idéntico de solución de problemas, el cual contiene todavía otro ciclo.

Raccoon sugiere un "Modelo del caos" que describa el "desarrollo del software como una extensión desde el usuario hasta el desarrollador y la tecnología".

A continuación se analizan algunos **modelos de procesos para la ingeniería del software**. Cada uno representa un intento por ordenar una actividad inherentemente caótica.

¹ Pressman Roger S. "Ingeniería del Software. Un enfoque práctico". Mc Graw Hill. 1989. 4ª Edición.

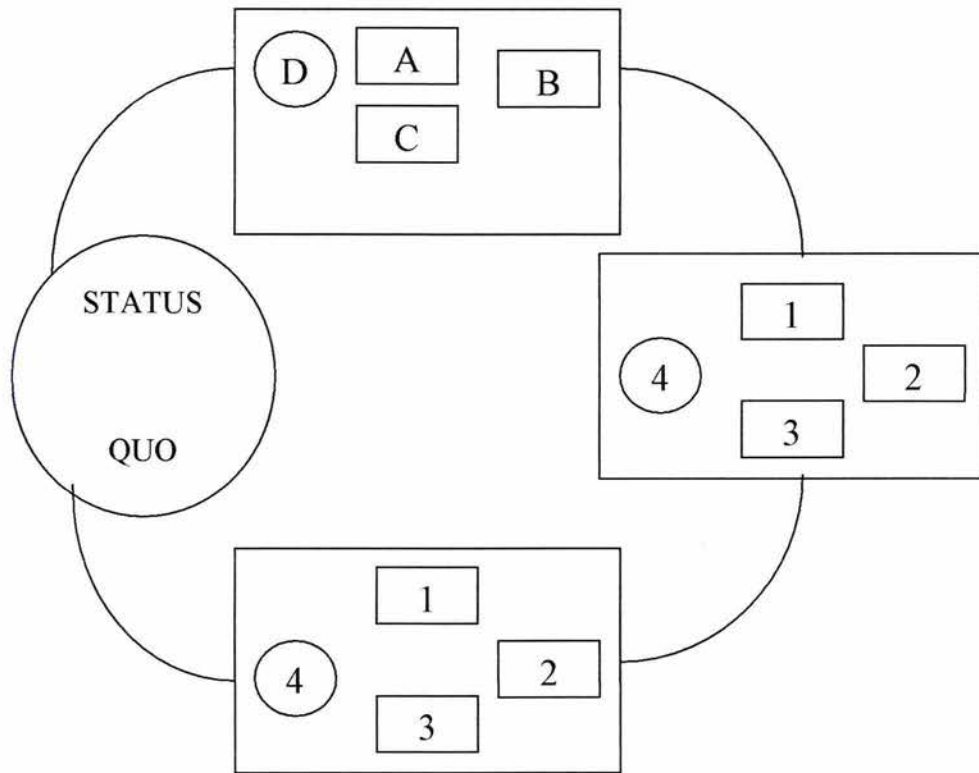


FIGURA 1: FASES DENTRO DE LAS FASES DE CICLO DE RESOLUCIÓN DE PROBLEMAS.

A, B Y C SON CICLOS COMO EL ORIGINAL. Y D, STATUS QUO.

1. Integración de problemas.
2. Desarrollo técnico.
3. Integración de soluciones.
4. Status Quo.

EL MODELO LINEAL SECUENCIAL

La figura 2 ilustra el modelo *lineal secuencial* para la ingeniería del software. Llamado algunas veces "ciclo de vida básico" o "modelo en cascada", el modelo lineal secuencial sugiere un enfoque sistemático, secuencial del desarrollo del software que comienza en un nivel de sistemas y progresa con el análisis, diseño, codificación, pruebas y mantenimiento. Modelado según el ciclo de ingeniería convencional, el modelo lineal secuencial acompaña a las actividades siguientes²:

² Ibid.

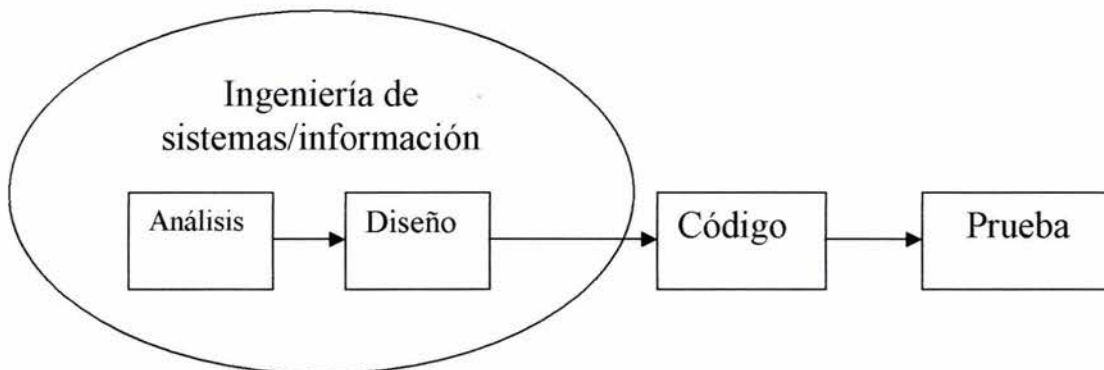


Figura 2. El modelo lineal secuencial.

Ingeniería y modelado de Sistemas/Información

Como el software siempre forma parte de un sistema más grande, el trabajo comienza estableciendo requisitos de todos los elementos del sistema y asignando al software algún subgrupo de estos requisitos. Esta visión del sistema es esencial cuando el software se debe interconectar con otros elementos como hardware, personas y bases de datos. La ingeniería y el análisis de sistemas acompañan a los requisitos que se recogen en el nivel estratégico de empresa y en el nivel del área de negocio.

Análisis de los requisitos del software

El proceso de reunión de requisitos se intensifica y se centra especialmente en el software. Para comprender la naturaleza del programa a construirse, el ingeniero de software debe comprender el dominio de información del software, así como la función requerida, comportamiento, rendimiento e interconexión. El usuario documenta y repasa los requisitos del sistema y del software.

Diseño

El diseño del software es realmente un proceso de varios pasos que se centra en cuatro atributos distintos de un programa: estructura de datos, arquitectura del software, representaciones de interfaz y detalle procedimental. El proceso de diseño traduce requisitos en una representación del software que se pueda evaluar por calidad antes de que comience la generación del código. Al igual que los requisitos, el diseño se documenta y se hace parte de la configuración del software.

Generación de código

El diseño se debe traducir a una forma legible por la máquina. El paso de generación de código lleva a cabo esta tarea. Si se lleva a cabo el diseño de una forma detallada, la generación de código se realiza mecánicamente.

Pruebas

Una vez que se ha generado el código, comienzan las pruebas del programa. El proceso de pruebas se centra en los procesos lógicos internos del software, asegurando que todas las sentencias se han comprobado, y en los procesos externos funcionales, es decir, la realización de las pruebas para la detección de errores y el sentirse seguro de que la entrada definida produzca resultados reales de acuerdo con los resultados requeridos.

Mantenimiento

El software indudablemente sufrirá cambios después de ser entregado al usuario. Se producirán cambios porque se han encontrado errores, porque el software debe adaptarse para acoplarse a los cambios de su entorno externo, o porque el usuario requiere mejoras funcionales o de rendimiento. El mantenimiento vuelve a aplicar cada una de las fases precedentes a un programa ya existente y no a uno nuevo.

El modelo lineal secuencial es el paradigma más antiguo y más extensamente utilizado en la ingeniería del software. Sin embargo, la crítica del paradigma ha puesto en duda su eficacia. **Entre los problemas que se encuentran** algunas veces en el modelo lineal secuencial se incluyen:

1. Los proyectos reales raras veces siguen el modelo secuencial que propone el modelo. Aunque el modelo lineal puede acoplar interacción, lo hace indirectamente. Como resultado, los cambios pueden causar confusión cuando el equipo del proyecto comienza.
2. Es difícil que el usuario exponga explícitamente todos los requisitos, es por ello que el modelo lineal secuencial tiene dificultades a la hora de acomodar la incertidumbre natural al comienzo de muchos proyectos.
3. El usuario debe tener paciencia. Una versión de trabajo del programa no estará disponible hasta que el proyecto esté muy avanzado. Un grave error puede ser desastroso si no se detecta hasta que se revisa el programa.
4. Los responsables del desarrollo del software siempre se retrasan innecesariamente. Bradac dijo que la naturaleza lineal de ciclo de vida clásico lleva a "estados de bloqueo" en el que algunos miembros del equipo del proyecto deben esperar a otros miembros del equipo para completar tareas dependientes. En efecto, el tiempo que se pasa esperando puede sobrepasar el tiempo que se emplea en el trabajo productivo. Los estados de bloqueo tienden a ser más importantes al principio y al final de un proceso lineal secuencial.

Cada uno de estos errores es real. Sin embargo, el paradigma del ciclo de vida clásico tiene un lugar definido e importante en el trabajo de la ingeniería del software. Proporciona una plantilla en la que se encuentran métodos para análisis, diseño, codificación, pruebas y mantenimiento. El ciclo de vida clásico sigue siendo el modelo de proceso más extensamente utilizado por la ingeniería de software. Pese a tener debilidades, es significativamente mejor que un enfoque hecho al azar para el desarrollo del software.

EL MODELO DE CONSTRUCCIÓN DE PROTOTIPOS

El paradigma de construcción de prototipos comienza con la recolección de requisitos. El desarrollador y el usuario encuentran y definen los objetivos globales para el software, identifican los requisitos conocidos y, las áreas del esquema en donde es obligatoria más definición. Entonces aparece un "diseño rápido". El diseño rápido se centra en una representación de esos aspectos del software que serán visibles para el usuario. El diseño rápido lleva a la construcción de un prototipo. El prototipo lo evalúa el usuario y lo utiliza para refinar los requisitos del software a desarrollar. La interacción ocurre cuando el prototipo satisface las necesidades del usuario, a la vez que permite que el desarrollador comprenda mejor lo que se necesita hacer³.

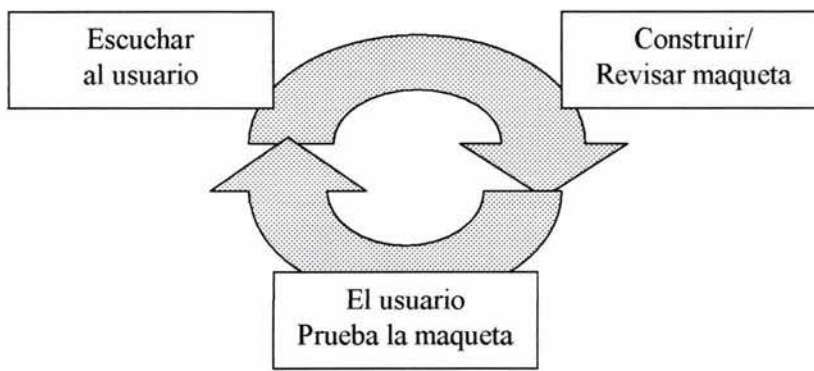


Figura 3. El paradigma de construcción de prototipos

Lo ideal sería que el prototipo sirviera como un mecanismo para identificar los requisitos del software. Si se construye un prototipo de trabajo, el desarrollador intenta hacer uso de los fragmentos del programa ya existentes o aplica herramientas que permiten generar rápidamente programas de trabajo.

Sin embargo, la construcción de prototipos también puede ser problemática por las razones siguientes:

1. Cuando se informa de que el producto se debe construir otra vez para que se puedan mantener los niveles altos de calidad, el usuario no lo entiende y pide que se apliquen "unos pequeños ajustes" para que se pueda hacer del prototipo un producto final.
2. El desarrollador a menudo hace compromisos de implementación para hacer que el prototipo funcione rápidamente. Se puede utilizar un sistema operativo o lenguaje de programación inadecuado simplemente porque está disponible y porque es conocido.

Aunque pueden surgir problemas, la construcción de prototipos puede ser un paradigma efectivo para la ingeniería del software. La clave es definir las reglas del juego al comienzo; es

³ Idid

decir, el usuario y el desarrollador se pueden poner de acuerdo en que el prototipo se construya para servir como un mecanismo de definición de requisitos. Entonces se descarta y se realiza la ingeniería del software con una visión hacia la calidad y la facilidad de mantenimiento.

EL MODELO DRA (RAPIDE APPLICATION DEVELOPMENT), DESARROLLO RÁPIDO DE APLICACIONES

Es un modelo de proceso del desarrollo del software lineal secuencial que enfatiza un ciclo de desarrollo extremadamente corto. Este modelo es una adaptación a "alta velocidad" del modelo lineal secuencial en el que se logra el desarrollo rápido utilizando un enfoque de construcción basado en componentes. Si se comprenden bien los requisitos y se limita el ámbito del proyecto, el proceso DRA permite al equipo de desarrollo crear un "sistema completamente funcional" dentro de periodos cortos de tiempo. Cuando se utiliza principalmente para aplicaciones de sistemas de información, el enfoque DRA comprende las siguientes fases⁴:

Modelado de gestión.

El flujo de información entre las funciones de gestión se modela de forma que responda a las siguientes preguntas: ¿Qué información conduce el proceso de gestión? ¿Qué información se genera? ¿Quién la genera? ¿Adónde va la información? ¿Quién la procesa?

Modelado de datos.

El flujo de información definido como parte de la fase de modelado de gestión se refina como un conjunto de objetos de datos necesarios para apoyar la empresa. Se definen las características de cada uno de los objetos y las relaciones entre estos objetos.

Modelado del proceso.

Los objetos de datos definidos en la fase de modelado de datos quedan transformados para lograr el flujo de información necesario para implementar una función de gestión. Las descripciones del proceso se crean para añadir, modificar, suprimir, o recuperar un objeto de datos.

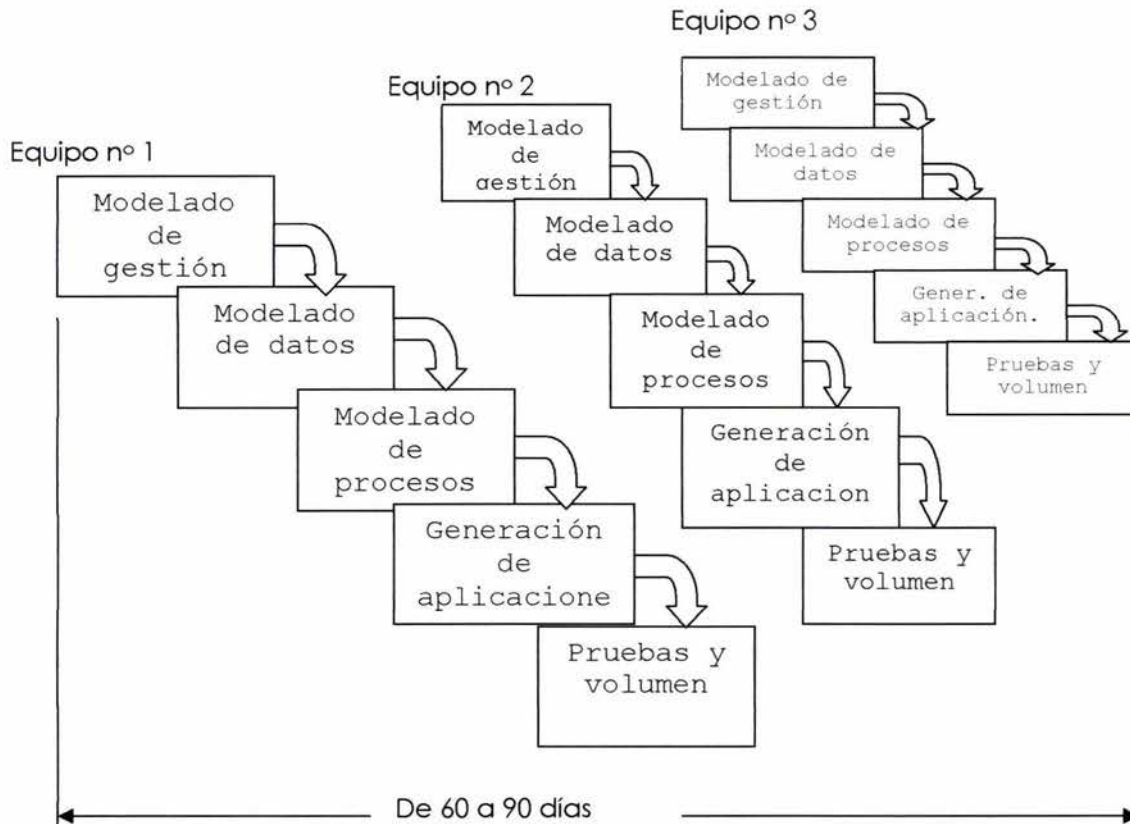
Generación de aplicaciones.

El DRA asume la utilización de técnicas de cuarta generación. En lugar de crear software con lenguajes de programación de tercera generación, el proceso DRA trabaja para volver a utilizar componentes de programas ya existentes o crear componentes reutilizables. En todos los casos se utilizan herramientas automáticas para facilitar la construcción del software.

Pruebas y entregas.

Como el proceso DRA enfatiza la reutilización, ya se han comprobado muchos de los componentes de los programas. Esto reduce tiempo de pruebas. Sin embargo, se deben probar todos los componentes nuevos y se deben ejercitar todas las interfaces a fondo.

⁴ Ibid



Obviamente, las limitaciones de tiempo impuestas en un proyecto DRA demandan "ámbito en escalas". Si una aplicación de gestión puede modularse de forma que permita completarse cada una de las funciones principales en menos de tres meses, es un candidato del DRA. Cada una de las fases pueden ser afrontadas por un equipo DRA diferente y ser integradas en un solo conjunto.

Al igual que todos los modelos de procesos, el modelo DRA tiene **inconvenientes**:

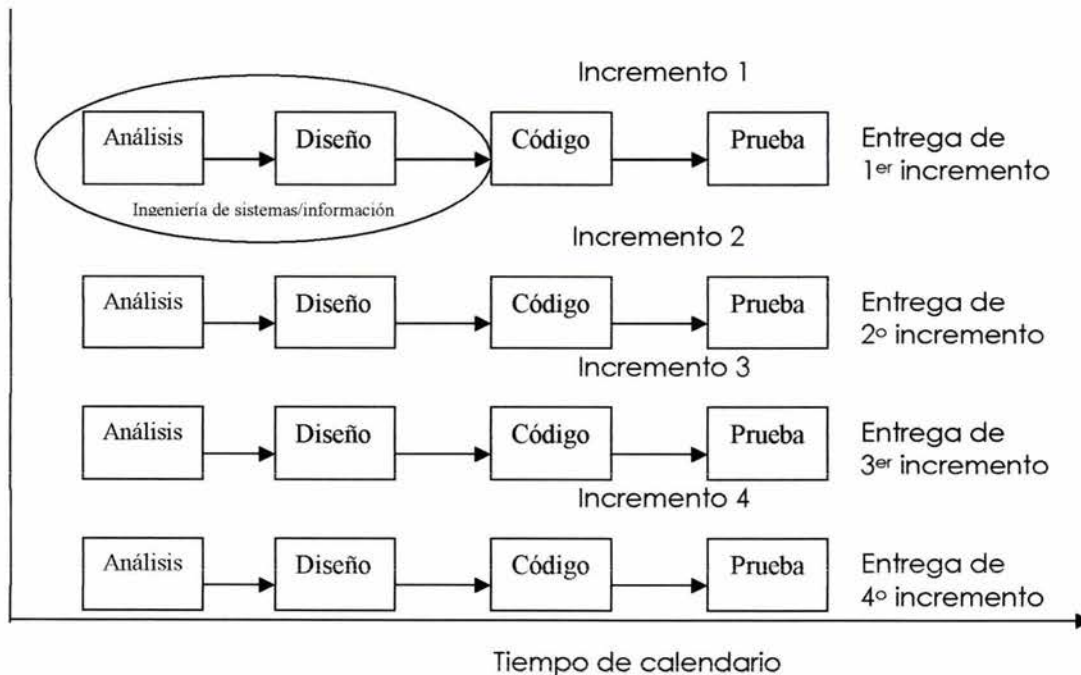
- Para proyectos grandes aunque por escalas, el DRA requiere recursos humanos suficientes como para crear el número correcto de equipos DRA.
- DRA requiere usuarios y desarrolladores comprendidos en las rápidas actividades necesarias para completar un sistema en un marco de tiempo abreviado. Si no hay compromiso, por ninguna de las partes constituyentes, los proyectos DRA fracasarán.

No todos los tipos de aplicaciones son apropiados para el DRA. Si un sistema no se puede modularizar adecuadamente, la construcción de los componentes necesarios para DRA será problemática. Si está en juego el alto rendimiento, y se va a conseguir el rendimiento convirtiendo interfases en componentes de sistemas, el enfoque DRA puede que no funcione. DRA no es adecuado cuando los riesgos técnicos son altos.

EL MODELO INCREMENTAL

El modelo incremental combina elementos del modelo lineal secuencial con la filosofía interactiva de construcción de prototipos. El modelo incremental aplica secuencias lineales de forma sorprendente de la misma forma que progresa el tiempo en el calendario. Cada secuencia lineal produce un "incremento" del software⁵.

Cuando se utiliza un modelo incremental, el primer incremento a menudo es un *producto esencial*. Es decir, se afrontan requisitos básicos, pero muchas funciones suplementarias quedan sin extraer. El usuario utiliza el producto central. Como un resultado de utilización y/o evaluación, se desarrolla un plan para el incremento siguiente. El plan afronta la modificación del producto central a fin de cumplir mejor las necesidades del usuario y la entrega de funciones, y características adicionales. Este proceso se repite siguiendo la entrega de cada incremento, hasta que se elabore el producto completo.



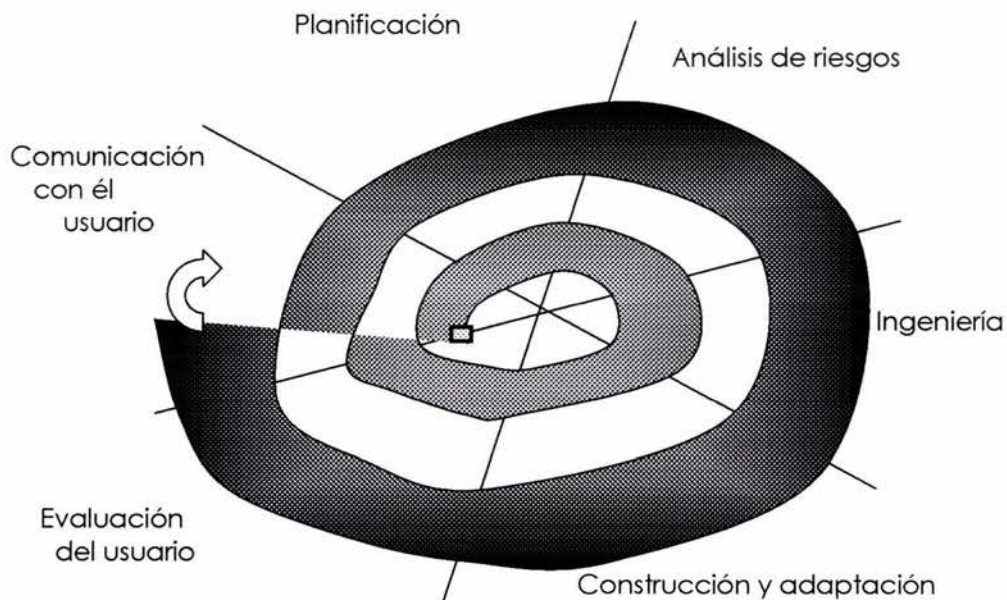
El modelo de proceso incremental, como la construcción de prototipos, es interactivo por naturaleza. Pero a diferencia de la construcción de prototipos, el modelo incremental se centra en la entrega de un producto operacional con cada incremento. Los primeros incrementos son versiones "desmontadas" del producto final, pero proporciona la capacidad que sirve al usuario y también proporciona una plataforma para la evaluación por parte del usuario.

El desarrollo incremental es particularmente útil cuando la dotación de personal no está disponible para una implementación completa en cuanto a la fecha límite de gestión que se haya establecido para el proyecto.

⁵ Ibid.

EL MODELO EN ESPIRAL

El modelo en espiral, propuesto originalmente por Boehm, es un modelo de proceso de software evolutivo que acompaña la naturaleza interactiva de construcción de prototipos con los aspectos controlados y sistemáticos del modelo lineal secuencial. Se proporciona el potencial para el desarrollo rápido de versiones incrementales del software. En el modelo espiral, el software se desarrolla en una serie de versiones incrementales. Durante las primeras iteraciones, la versión incremental podría ser un modelo en papel o un prototipo. Durante las últimas iteraciones, se producen versiones cada vez más completas de ingeniería del sistema⁶.



El modelo en espiral se divide en un número de *actividades estructurales*, también llamadas *regiones de tareas*. Generalmente, existen entre tres y seis regiones de tareas:

- **Comunicación con el usuario.** Tareas establecidas para entablar comunicación entre él desarrollados y el usuario.
- **Planificación.** Tareas requeridas para definir recursos, el tiempo y otras informaciones relacionadas con el proyecto.
- **Análisis de riesgos.** Tareas requeridas para evaluar riesgos técnicos y de gestión.
- **Ingeniería.** Tareas requeridas para construir una o más representaciones de la aplicación.
- **Construcción y adaptación.** Tareas requeridas para construir, probar, instalar y proporcionar soporte al usuario.

⁶ Ibid.

- **Evaluación del usuario.** Tareas requeridas para obtener la reacción del usuario según la evaluación de las representaciones del software creadas durante la etapa de ingeniería e implementada durante la etapa de instalación.

Cuando empieza este proceso evolutivo, el equipo de ingeniería del software gira alrededor de la espiral en la dirección de las agujas del reloj, comenzando por el centro. El primer circuito de la espiral produce el desarrollo de una especificación de productos; los pasos siguientes en la espiral se podrían utilizar para desarrollar un prototipo y progresivamente versiones más sofisticadas del software. Cada paso de la región de planificación produce ajustes en el plan del proyecto. El costo y la planificación se ajustan según la reacción ante la evaluación del cliente. Además, el gestor del proyecto ajusta el número planificado de iteraciones requeridas para completar el software.

El modelo en espiral puede adaptarse y aplicarse a lo largo de la vida del software de la computadora.

En la figura se define un eje de *punto de entrada en el proyecto*. Un proyecto de desarrollo de conceptos comienza en el centro de la espiral y continuará hasta que se complete el desarrollo del concepto. Si el concepto se va a desarrollar dentro de un producto real, el proceso procede a través del cubo siguiente y se inicia un nuevo proyecto de desarrollo. El producto nuevo evolucionará a través de iteraciones alrededor de la espiral siguiendo el camino que limita la región algo más brillante que el centro. Un flujo de proceso similar aparece para otros tipos de proyectos.

En esencia, la espiral, cuando se caracteriza de esta forma permanece operativa hasta que el software se retira.

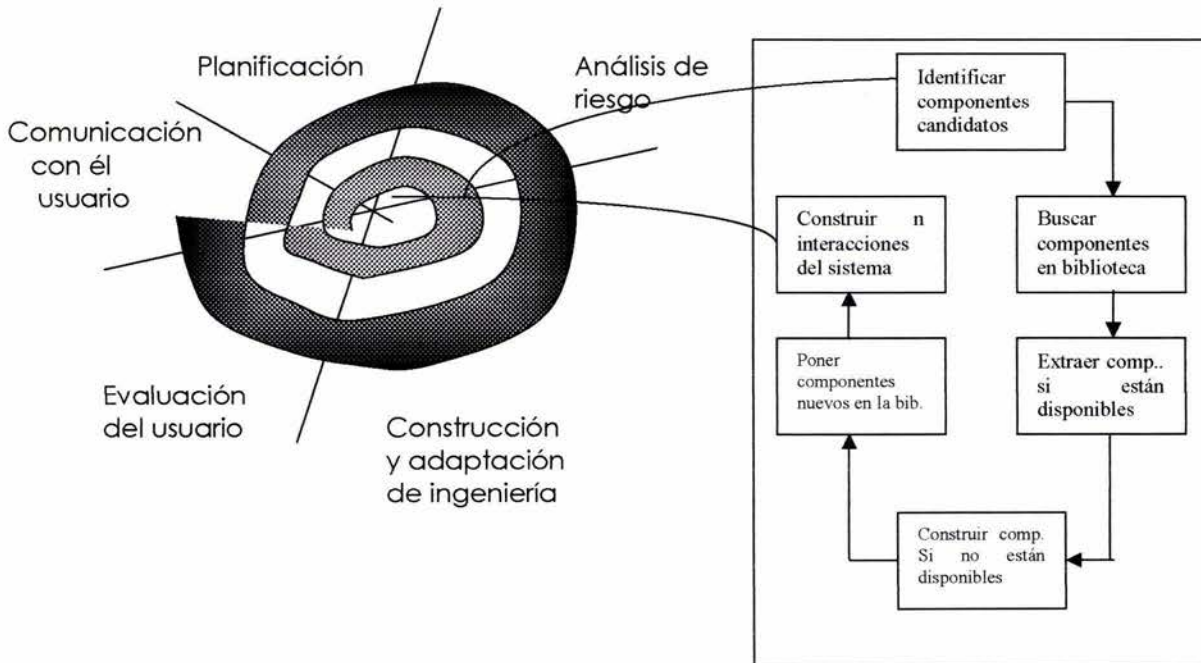
El modelo en espiral es un enfoque realista del desarrollo de sistemas y de software a gran escala; utiliza la construcción de prototipos como mecanismo de reducción de riesgos, pero lo que es más importante, permite a quien lo desarrolla aplicar el enfoque de construcción de prototipos en cualquier etapa de evolución del producto. Mantiene el enfoque sistemático de los pasos sugeridos por el ciclo de vida clásico, pero lo incorpora al marco de trabajo interactivo que refleja de forma más realista el mundo real. El modelo en espiral demanda una consideración directa de los riesgos técnicos en todas las etapas del proyecto, y si se aplica adecuadamente, debe reducir los riesgos antes de que se conviertan en problemáticos.

Requiere una considerable habilidad para la evaluación del riesgo, y cuenta con esta habilidad para el éxito. Si un riesgo importante no es descubierto y gestionado, indudablemente surgirán problemas. Finalmente, el modelo en sí mismo es relativamente nuevo y no se ha utilizado tanto como los paradigmas lineales secuenciales o de construcción de prototipos. "Todavía tendrán que pasar muchos años antes de que se determine con absoluta certeza la eficacia de este nuevo e importante paradigma".

EL MODELO DE ENSAMBLAJE DE COMPONENTES

El paradigma de orientación a objetos enfatiza la creación de clases que encapsulan tanto los datos como los algoritmos que se utilizan para manejar datos. Si se diseñan e implementan

ra manejar datos. Si se diseñan e implementan adecuadamente, las clases orientadas a objetos son reutilizables por las diferentes aplicaciones y arquitecturas de sistemas basados en computadora⁷.



El modelo de ensamblaje de componentes incorpora muchas de las características del modelo en espiral. Es evolutivo por naturaleza y exige un enfoque interactivo para la creación del software. Sin embargo, el modelo ensamblador de componentes configura aplicaciones desde componentes preparados de software.

La actividad de la ingeniería comienza con la identificación de las clases candidatas. Esto se lleva a cabo examinando los datos que se van a manejar por parte de la aplicación y el algoritmo que se va a aplicar para conseguir el tratamiento. Los datos y los algoritmos correspondientes se empaquetan en una clase.

Las clases creadas en los proyectos de ingeniería del software anteriores se almacenan en una *biblioteca de clases* o *depósito*. Una vez identificadas las clases candidatas, la biblioteca de clases se examina para determinar si estas clases ya existen. En caso de que así fuera, se extraen de la biblioteca y se vuelven a utilizar. Si una clase candidata no reside en la biblioteca, se aplican los métodos orientados a objetos. Se compone así la primera interacción de la aplicación a construirse, mediante las clases extraídas de la biblioteca y las clases nuevas construidas para cumplir las necesidades únicas de la aplicación. El flujo del proceso vuelve a la espiral y volverá a introducir por último la interacción ensambladora de componentes a través de la actividad de ingeniería.

⁷ Ibid.

EL MODELO DE DESARROLLO CONCURRENTRE

Se puede representar en forma de esquema como una serie de actividades técnicas importantes, tareas, y estados asociados a ellas.

La actividad –análisis- se puede encontrar en uno de los estados destacados anteriormente en cualquier momento dado. De forma similar, otras actividades se pueden representar de forma análoga⁸.



Todas las actividades existen concurrentemente, pero residen en estados diferentes.

El modelo de proceso concurrente define una serie de acontecimientos que dispararán transiciones de estado a estado para cada una de las actividades de la ingeniería del software.

Se utiliza a menudo como el paradigma de desarrollo de aplicaciones cliente/servidor. Un sistema cliente/servidor está formado de un conjunto de componentes funcionales. Cuando se aplica a cliente/servidor, el modelo de proceso concurrente define actividades en dos dimensiones: una *dimensión de sistemas* y una *dimensión de componentes*. Los aspectos del nivel de sistemas se afrontan mediante tres actividades: *diseño*, *embalaje* y *uso*. La dimensión de componentes se afronta con dos actividades: *diseño* y *realización*. La concurrencia se logra de dos formas: (1) las actividades de sistemas y de componentes ocurren simultáneamente y pueden modelarse con el enfoque orientado a objetos; (2) una aplicación cliente/servidor

⁸ Ibid.

típica se implementa con muchos componentes, cada uno de los cuales se pueden diseñar y realizar concurrentemente.

En realidad, el modelo de proceso concurrente es aplicable a todo tipo de desarrollo de software y proporciona una imagen exacta del estado actual de un proyecto. En vez de confinar las actividades de ingeniería del software a una secuencia de sucesos, define una red de actividades. Todas las actividades de la red existen simultáneamente con otras. Los sucesos generados dentro de una actividad dada o en algún otro lugar en la red de actividad inician las transiciones entre los estados de una actividad.

EL MODELO DE MÉTODOS FORMALES

El modelo de métodos formales acompaña a un conjunto de actividades que conducen a la especificación matemática del software de computadora. Los métodos formales permiten que un ingeniero del software especifique, desarrolle y verifique un sistema basado en computadora aplicando una notación rigurosa y matemática⁹.

Cuando se utilizan métodos formales durante el desarrollo, proporcionan un mecanismo para eliminar muchos de los problemas que son difíciles de superar con paradigmas de la ingeniería del software. La ambigüedad, lo incompleto y la inconsistencia se descubren y se corrigen más fácilmente, no mediante una revisión a propósito para el caso, sino mediante la aplicación del análisis matemático. Cuando se utilizan métodos formales durante el diseño, sirven como base para la verificación de programas y por consiguiente permiten que el ingeniero del software descubra y corrija errores que no se pudieron detectar de otra manera.

Aunque todavía no hay un enfoque establecido, los modelos de métodos formales ofrecen la promesa de un software libre de defectos. Sin embargo, se ha hablado de una gran preocupación sobre su aplicabilidad en un entorno de gestión:

1. El desarrollo de modelos formales actualmente es bastante caro y lleva mucho tiempo.
2. Se requiere un estudio caro porque pocos responsables del desarrollo de software tienen los antecedentes necesarios para aplicar métodos formales.
3. Es difícil utilizar los modelos como un mecanismo de comunicación con usuarios que no tienen muchos conocimientos técnicos.

DIAGRAMAS DE CONTEXTO Y FLUJO DE DATOS

Introducción

Podemos hacer uso de herramientas gráficas que nos pueden ilustrar el manejo de información y procesos en el análisis del ciclo de vida de software. Estas herramientas se conocen como **diagramas de contexto y de flujo de datos**.

⁹ Ibid.

Diagramas de contexto.

En una de las primeras etapas de la obtención de requerimientos y del proceso de análisis se deben definir los límites del sistema. Una vez que se han tomado las decisiones sobre los límites del sistema, una parte de la actividad de análisis es la definición de ese contexto y las dependencias que un sistema tiene con su entorno. Por lo regular, el primer paso en esta actividad es la producción de un modelo arquitectónico sencillo que muestre la entrada del proceso, el proceso general para la solución del problema y el resultado esperado con dicho procesamiento.

Diagrama de flujo de datos.

Es una forma intuitiva de mostrar la manera en los datos son procesados en el sistema existente. La notación utilizada en estos modelos representa el procesamiento funcional, los almacenes y los movimientos de datos entre las funciones.

Los modelos de flujo de datos se utilizan para mostrar cómo fluyen los datos a través de una secuencia de pasos de procesamiento. Los datos se transforman en cada paso antes de moverse a la siguiente etapa. Si los diagramas de flujo de datos se utilizan para documentar un diseño de software, estos pasos de funcionamiento o transformaciones se representan por funciones en los programas¹⁰.

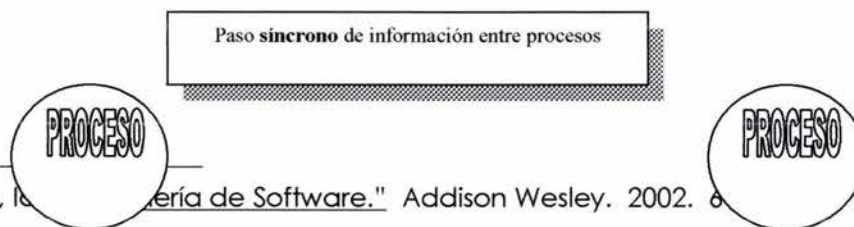
La nomenclatura utilizada en la elaboración de estos diagramas es la siguiente:

REPRESENTACIÓN GRÁFICA DEL DIAGRAMS DE FLUJO DE DATOS.

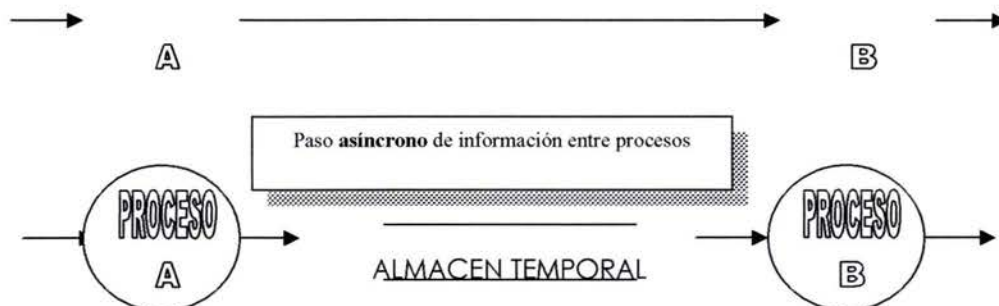
Conexiones permitidas.

DESTINO FUENTE	PROCESO	ALMACEN EXTERNA	ENTIDAD
PROCESO	SI	SI	SI
ALMACEN	SI	NO	NO
ENTIDAD			
EXTERNA	SI	NO	NO

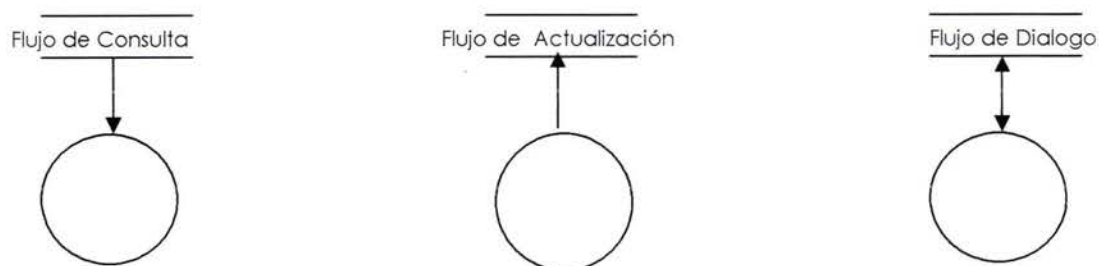
Formas de paso de datos entre procesos.



¹⁰ Sommerville, I. "Ingeniería de Software." Addison Wesley. 2002. 6



Conexiones entre procesos y almacenes.



SELECCIÓN DE LA METODOLOGÍA A UTILIZAR

De los modelos de procesos que se describieron anteriormente, y de acuerdo con las características de nuestras necesidades, la metodología que se aplicará tiene las siguientes etapas:

Primera etapa: Ingeniería y modelado de sistemas en el cual el solicitante expone la problemática en la que se encuentra y los requisitos que se necesita cubrir para las inscripciones a los laboratorios que coordina la DIMEI¹¹ en el departamento de Ingeniería Mecánica y establecidas por el plan de estudios vigente; contemplando las siguientes necesidades del Departamento:

1. Recibir la información obtenida durante el proceso de inscripciones ordinarias por parte de la USECAD.
2. Realizar las altas, bajas y cambios de grupo de los alumnos que cursan los laboratorios en el departamento de Ingeniería Mecánica.
3. Enviar la información definitiva de los movimientos realizados durante las inscripciones complementarias a la USECAD.

¹¹ La DIMEI (División de Ingeniería Mecánica e Industrial) coordina las carreras de Ingeniería Mecánica, Ingeniería Industrial e Ingeniería Mecatrónica.

4. Una vez realizados los trámites, generar listas de asistencia, listas definitivas de grupo, listas de calificaciones y credenciales para el acceso a los laboratorios del Departamento de Ingeniería Mecánica.
5. Contar con la posibilidad de almacenar durante dos semestres las calificaciones de los alumnos para que estos puedan validarlas, en caso de no aprobar la teoría correspondiente.

Segunda etapa: Análisis de los requerimientos del software el cual nos lleva a realizar un análisis de las herramientas comerciales que nos permitan cubrir las necesidades del usuario; tomando en cuenta el manejo de información, transferencia de la misma, desarrollo de aplicaciones y consultas a base de datos entre otras.

Tercera etapa: Diseño, este apartado permitirá estructurar de forma adecuada la propuesta a desarrollar, es decir, manejo de la información, diseño de la interfaz gráfica y el funcionamiento de la misma.

Cuarta etapa: Generación de código, desarrollo de la propuesta utilizando las herramientas comerciales seleccionadas para la construcción de interfaces, conexión a bases de datos a través de una red y creación de la base de datos.

Quinta etapa: Pruebas, una vez terminada la propuesta se indicará algunas pruebas que verifiquen su funcionamiento.

Sexta etapa: Mantenimiento, En esta etapa se realizarán los cambios que la propuesta pudiera requerir para mantener su buen funcionamiento.

CAPÍTULO 3 MANEJO DE BASES DE DATOS

Las bases de datos constituyen una parte integral y fundamental del sistema de información y tienen su razón de ser en la misma existencia de éste.

Adoración de Miguel y Mario Piattini.

En este capítulo definiremos algunos conceptos que serán de utilidad durante el desarrollo de una base de datos para nuestra propuesta.

BASE DE DATOS

Conjunto de datos interrelacionados con independencia física y lógica, consistentes, íntegros y con redundancia controlada, almacenados más o menos permanentemente en una computadora, de forma que:

- a) Los datos son compartidos por diferentes usuarios y programas de aplicación; existe un mecanismo común para inserción, actualización, borrado y consulta de los datos.
- b) Tanto los usuarios finales como los programas de aplicación no necesitan conocer los detalles de las estructuras de almacenamiento.

Objetivos de una base de datos.

- Proteger el valor de los datos.
- Hacer que las fuentes de datos sean responsables de los cambios de las necesidades de información.
- Permitir que la organización que procesa datos logre un mejor control y seguimiento de sus planes de negocios así como de sus metas.
- Reducir los costos de optimización del desempeño.

Costos

Los costos de establecer y operar en un ambiente de bases de datos incluyen:

- Costos de la tecnología DBMS.
- Costos de operación de la Base de Datos.
- Costos de Conversión de los datos y la lógica.
- Costos de Planeación.
- Costos de Riesgo.

Ventajas de las bases de datos.

- *Reduce la redundancia.* En sistemas que no usan bases de datos cada aplicación tiene sus propios archivos privados. Esto a menudo origina enorme redundancia en los datos almacenados, así como desperdicio resultante del espacio de almacenamiento.

- Evita la inconsistencia (al menos en cierta medida). Esto sucede cuando al hacer una petición no obtenemos el resultado correcto, por ejemplo, si solicitamos el teléfono de alguna persona podemos tener como resultado un número incorrecto.
- Los datos se comparten. En algunas ocasiones las aplicaciones existentes pueden compartir los datos de la base de datos, y también es factible desarrollar nuevas aplicaciones que operen con los mismos datos almacenados.
- Pueden hacerse cumplir normas establecidas: Con un control central de la base de datos, el administrador de la base de datos puede garantizar que se cumplan todas las formas aplicables a la representación de los datos. Las normas aplicables pueden comprender la totalidad o parte de lo siguiente: normas de la compañía, de instalación, departamentales, industriales, nacionales o internacionales. Es muy deseable unificar los formatos de los datos almacenados como ayuda para el intercambio o migración de datos entre sistemas.
- Pueden aplicarse restricciones de seguridad: Al tener jurisdicción completa sobre los datos de operación, el administrador de la base de datos puede:
 - a. asegurar que el único medio de acceder a la base de datos sea a través de canales establecidos y, por tanto,
 - b. definir controles de autorización para que se apliquen cada vez que se intente el acceso a datos sensibles.
- Conservan la integridad de los datos. Se refiere a la validez de datos.
- Pueden equilibrarse los requerimientos contradictorios. Cuando conoce los requerimientos globales de la empresa, en contraste con los requerimientos de cualquier usuario individual, el administrador de la base de datos puede estructurar el sistema de bases de datos para brindar un servicio que sea "el mejor para la empresa" en términos globales.
- Es compacto. No hacen falta archivos de papeles que pudieran ocupar mucho espacio.
- Es rápido. La máquina puede obtener y modificar datos con mucha mayor velocidad que un ser humano, así es posible satisfacer con rapidez consultas de casos particulares, sin necesidad de búsquedas visuales o manuales que requieren mucho tiempo.
- Es menos laborioso. Se elimina gran parte del trabajo de mantener archivos a mano, las tareas mecánicas siempre serían mejor realizadas por las máquinas.
- Es actual. Se dispone en cualquier momento de información precisa y al día.

Independencia lógica de los datos.

El programa de aplicación puede cambiar sin afectar a los datos almacenados.

Independencia física de los datos

Las aplicaciones permanecen inalteradas sin importar los cambios efectuados en el almacenamiento o en los métodos de acceso.

BASES DE DATOS RELACIONALES (O MODELO DE DATOS RELACIONAL).

E. F. Codd desarrolló en IBM – San José (California) el modelo de datos relacional. Este modelo está basado en conceptos muy sencillos teniendo asociada la teoría de normalización de relaciones que tiene por objeto la eliminación de los comportamientos anómalos de las relaciones durante los procesos de manejo de la información que representan y la eliminación de redundancias superfluas, facilitando así, la comprensión del esquema en cuanto a las relaciones semánticas existentes entre los objetos del dominio del problema.

Una base de datos relacional puede ser estructurada físicamente de múltiples formas, aunque es lógico, la representación física deberá satisfacer y representar, de alguna forma, las relaciones y restricciones lógicas del esquema relacional.

El modelo relacional propone una representación de la información que:

- Origine esquemas que representen fielmente la información, los objetos y relaciones entre ellos existente en el dominio del problema.
- Pueda ser entendida fácilmente por los usuarios que no tienen una preparación previa en esta área.
- Haga posible ampliar el esquema de la base de datos sin modificar la estructura lógica existente y, por tanto, sin modificar los programas de aplicación.
- Permita la máxima flexibilidad en la formulación de los interrogantes previstos, y no previstos, sobre la información mantenida en la base de datos.

Como cualquier modelo de datos, el modelo relacional, introduce su propia terminología para nominar los objetos y elementos utilizados por él para representar el dominio de la información; está soportado sobre una teoría de igual nombre basada en los principios de la teoría general de conjuntos. Si bien una relación representa a un conjunto, y como tal puede y debe ser considerada, no todos los conjuntos pueden ser considerados en un esquema relacional para satisfacer en la base de datos los siguientes objetivos:

1. No-existencia de redundancias superfluas, aminorando el espacio requerido para el almacenamiento de la información y, por tanto, reduciendo posibles problemas de integridad en la información almacenada en la base de datos.
2. Aumentar el desempeño de las operaciones de actualización de la base de datos.
3. Representar de forma coherente los objetos y relaciones existentes en el dominio del problema y cuya información es almacenada en la base de datos.
4. Aumentar el desempeño y garantizar la fiabilidad de las interrogaciones sobre la información mantenida en la base de datos.

Para satisfacer estos objetivos, las relaciones que forman parte de un esquema relacional deben satisfacer una serie de reglas que restringen el universo de relaciones/conjuntos que pueden ser considerados en un esquema relacional. Estas reglas se conocen como las **reglas de Codd**.

Reglas de Codd

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

Regla 1. Toda la información dentro de una base de datos relacional se representa de una manera explícita a nivel lógico y exactamente de una sola manera, como valores de una tabla.

Regla 2. Se garantiza que todos y cada de los datos (valor atómico) en una base de datos relacional pueden ser leídos recurriendo a una combinación de nombre de la tabla, valor de la llave primaria y nombre de columna.

Regla 3. En un DBMS totalmente relacional se soportan los valores nulos (que son distintos de una cadena vacía o de una cadena con caracteres en blanco o de cero o cualquier otro número). Para representar información faltante o no aplicable de una forma consiente independientemente del tipo de dato.

Regla 4. La descripción de la base de datos se representa en el nivel lógico de la misma forma que los datos ordinarios, de tal suerte que los usuarios autorizados puedan aplicar el mismo lenguaje relacional para consultarla, que aquel que emplean con sus datos habituales.

Regla 5. Deberá contener un sublenguaje de datos completo que permita

- Definición de datos.
- Definición de vistas.
- Manipulación de datos.
- Restricciones de integridad (manejo).
- Autorización.
- Inicio y fin de una transacción.

Regla 6. Todas las vistas que teóricamente sean actualizadas por medio del sistema.

Regla 7. La posibilidad de manejar una relación base o una relación derivada como un solo operador se aplica a la lectura, inserción, modificación y eliminación de datos.

Regla 8. Los programas aplicativos y la actividad en terminales no deberán ser afectados por cambios en el almacenamiento físico de los datos o en los métodos de acceso.

Regla 9. Los programas aplicativos y la actividad en terminales no deberán ser afectados por cambios de cualquier tipo que preserven la información y que teóricamente permitan la afectación en las tablas base.

Regla 10. Las restricciones de integridad de una base de datos deberán definirse con el mismo sublenguaje de datos relacional y deberá almacenarse en el catálogo (tablas), no en los programas aplicativos.

Regla 11. Un DBMS relacional tiene independencia de distribución.

Regla 12. Si un sistema relacional tiene un lenguaje de bajo nivel (que opere un registro cada vez), ese lenguaje no deberá poder emplearse para subvertir las reglas de integridad y las restricciones expresadas en el lenguaje relacional de alto nivel.

SISTEMA MANEJADOR DE BASES DE DATOS

Para que la información pueda ser almacenada y el acceso a la misma satisfaga las características exigidas en una base de datos y ser denominada como tal, es necesario que exista un software que sea capaz de llevar a cabo tal labor. A éste es a lo que se le denomina **DBMS** (DataBase Management System en inglés); o Sistema Manejador (o de Gestión) de Bases de Datos.

Así el DBMS es el software que controla la organización, almacenamiento, recuperación, seguridad, integridad y manejo de datos en una base de datos haciendo uso de algún modelo de datos. Acepta pedidos de datos desde un programa de aplicación o cliente y le ordena al sistema operativo transferir los datos apropiados. Cuando se un DBMS, los sistemas pueden ser cambiados con mayor facilidad a medida que cambien los requerimientos de la organización y agregar nuevas categorías de datos sin dañar el sistema existente.

Estructura General de un DBMS.

Los DBMS están particionados en varios componentes de software (módulos) cada uno de los cuáles le es asignada una operación específica. Algunas de las funciones de los DBMS son soportadas por sistema operativo subyacente. Sin embargo, el sistema operativo provee únicamente servicios rudimentarios y el DBMS debe ser construido sobre ello. Así, el diseño de un DBMS debe tomar en cuenta la interfaz entre el DBMS y el sistema operativo.

Componentes del Sublenguaje Empleado por el DBMS.

- **Lenguaje de Definición de Datos, LDD (Data Definition Language, DDL).**

Permite la definición de o descripción de los objetos de la base de datos. Puede usarse para crear, alterar o borrar relaciones (tablas), vistas, restricciones de integridad (por ejemplo, llaves primarias y llaves foráneas), tipos de datos, índices, reglas, defaults, vistas, triggers, procedimientos almacenados, etc.

El DBMS debe ser capaz de aceptar definiciones de datos (esquemas externos, el esquema conceptual, el esquema interno y todas las correspondientes asociadas) en versión fuente y convertirlas en la versión objeto apropiada.

- **Lenguaje de Manipulación de Datos, LMD (Data Manipulation Language, DML).**

Apoya el manejo o procesamiento de los objetos de la base de datos. Puede usarse para leer (consultar), modificar, borrar, o agregar tuplas (renglones) a las relaciones existentes. Una de las primeras funciones de los DBMS es la de soportar un DML en el cual el usuario pueda formular comandos que permitan manipular datos. Los DML se distinguen por sus sublenguajes de recuperación subyacentes:

- a. Lenguajes procedurales: Se tratan los registros individualmente.
- b. Lenguajes no procedurales: Se opera sobre un conjunto de registros.

El DBMS debe ser capaz de atender las solicitudes del usuario para extraer, y quizá poner al día los datos que ya existen en la base de datos, o para agregar en ella datos nuevos. Dicho de otro modo, el DBMS debe incluir un componente procesador de lenguaje de manipulación de datos. Las solicitudes en el DML pueden ser:

Planeada: Es aquella cuya necesidad se previó mucho tiempo antes de que tuviera que ejecutarse por primera vez.

No planeada: Es una consulta *ad hoc*, es decir, una solicitud cuya necesidad no se previó sino que surgió de improviso.

Un ejemplo es SQL: Las instrucciones insert, update, select, delete, etc.

- **Diccionario de Datos, DD (Data Dictionary, DD).**

Es una base de datos que contiene "datos acerca de datos". En particular, todos los diversos esquemas (externo, conceptual e interno), se almacenan físicamente en el diccionario, tanto en forma fuente como en forma objeto. Entre los tipos de información que el sistema debe de almacenar están:

- Los nombres de las relaciones.
- Los nombres de los atributos de cada relación.
- Los dominios de los atributos.
- Los nombres de las vistas definidas en la base de datos y la definición de esas vistas.
- Las restricciones de integridad de cada relación (por ejemplo, las restricciones de clave).

Van a tener la información de todos los objetos de la base de datos. Sus principales funciones son las siguientes:

- Describe todos los elementos en el sistema (flujo de datos, procesos).
- Los elementos se centran en los datos y en la forma en que están estructurados.
- Comunica los mismos significados para todos los elementos del sistema.
- Documenta las características del sistema.
- Facilita el análisis de los detalles para evaluar las características y determinar cómo deben realizarse los cambios.
- Localiza errores y omisiones en el sistema.

Además de esto es recomendable que en la mayoría de los sistemas se conserven los siguientes datos.

- Nombre de los usuarios autorizados.
- Información contable acerca de los usuarios.

En los sistemas que utilizan estructuras altamente sofisticadas para almacenar relaciones, pueden conservarse datos estadísticos y descriptivos acerca de las relaciones:

- Numero de tuplas de cada relación.
- Método de almacenamiento utilizado para cada relación.

Es importante almacenar la información de los índices de cada una de las relaciones.

- Nombre del índice.
- Nombre de la relación que se indexa.
- Atributos sobre los que está el índice.
- Tipo de índice.

Toda esta información constituye, de hecho, una base de datos en miniatura. Generalmente es preferible almacenar los datos acerca de la base de datos en la misma base de datos.

Ventajas

- a. Acceso eficiente de datos.
- b. Evita redundancia.
- c. Portabilidad.
- d. Facilita el uso compartido de los datos.
- e. Facilita y reduce el tiempo de desarrollo de las aplicaciones.

- f. Acceso en línea y/o en batch (por lotes).
- g. Confiabilidad de los datos.

Desventajas

- a. Vulnerables
- b. Se pierde el sentido de propiedad de los datos.
- c. Es difícil percibir errores pequeños.

- **Lenguaje de Control de Datos, LCD (Data Control Language, DCL)**

Permite la definición de los usuarios de la base de datos. Pueden usarse para crear, alterar o eliminar permisos de acceso y manipulación a la base de datos por diferentes usuarios.

EL ENFOQUE RELACIONAL.

Diseño de bases de datos.

Un DBMS utiliza un modelo de datos para definir la estructura fundamental de las bases de datos. Consta de tres modelos (modelos conceptual, lógico y físico). Es una representación abstracta de los datos; define la forma en que los datos elementales son organizados y relacionados.

Modelo Conceptual

En el modelo conceptual se va a realizar un análisis de datos, así mismo se consideran las aplicaciones existentes y potenciales. Define y modela aspectos importantes de la información que el negocio necesita saber o tener. Se ejecuta durante las fases de análisis y estrategia del ciclo de desarrollo del sistema y debe basarse en la visión del usuario acerca del proceso a automatizar y los datos implicados. Para ello es posible auxiliarse de las siguientes herramientas:

- Pseudocódigo
- Diagramas de flujo (estructuras de control)
- Diagramas de flujo de datos
- Diagramas entidad relación
- Diagramas de estructura (módulos)
- Diseño orientado a objetos
- Manuales de procedimientos
- Manuales de organización
- Árboles de decisión
- Tablas de decisión

Son las unidades básicas para construir cualquier base de datos que se ajuste al modelo.

Modelo Lógico

Requerimientos y procedimientos impuestos por un DBMS. Colección de reglas generales de integridad, las cuales limitan el conjunto de casos de esos tipos de objetos que pueden aparecer e forma legal en cualquier base de datos que se ajuste al modelo.

Modelo Físico

Colección de operadores aplicables a casos de objetos para obtener información y para otros propósitos. Especifica como se almacenarán los datos; el espacio que será ocupado; métodos de acceso rápido de datos, etc.

DIAGRAMA ENTIDAD-RELACIÓN

La representación del modelo conceptual más utilizado en el análisis de los problemas es aquella que está basada en el modelo de datos denominado **Entidad-Relación (E-R)**. Estos modelos a menudo reflejan la comprensión que el usuario final tiene del sistema. También contribuyen directamente a la identificación de objetos (los datos que fluyen) y a la identificación de las operaciones sobre esos objetos.

Los diferentes tipos de modelos se basan en varios enfoques de abstracción. Un modelo de flujo de datos se concentra (por ejemplo) en el flujo de éstos y en las transformaciones funcionales de los mismos. Omite detalles de las estructuras de los datos. En contraste, un modelo entidad-relación es para documentar los datos del sistema y su relación sin tomar en cuenta las funciones de éste. De manera similar es el funcionamiento del modelo de contexto.

El modelo entidad-relación fue propuesto por Peter Chen a mediados de los años setenta para la representación conceptual de los problemas y como un medio para representar la visión de un sistema de forma global.

El modelo Entidad-Relación (E-R), está soportado en la representación de los datos haciendo uso de gráficas, tablas y llaves. Mediante un conjunto de símbolos y haciendo uso de un conjunto reducido de reglas, son representados los elementos que forman parte del sistema y las relaciones existentes entre ellos, siendo estos elementos descritos mediante un pseudolenguaje basado en una gramática sencilla. El modelo E-R propone el uso de tablas bidimensionales para la representación particular de cada uno y, por tanto, de los conjuntos de elementos particulares y sus relaciones existentes en el sistema.

Antes de empezar a describir el modelo E-R es necesario introducir una serie de conceptos básicos que son utilizados por el mismo. Así, se define:

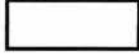





- Conjunto. Se denomina conjunto, en este contexto, al igual que en la *Teoría Clásica de Conjuntos*, a la agregación de una serie de objetos elementales mediante una función de pertenencia. La función de pertenencia caracteriza a los elementos como parte del conjunto, no siendo importante el orden de los elementos dentro del conjunto, ni duplicación de los mismos.
- Relación. Se denomina *relación* a un conjunto que representa una correspondencia entre dos o más conjuntos. Una relación es, por tanto, un nuevo conjunto en que cada elemento está formado por la agregación de los elementos de los conjuntos individuales que intervienen en la relación.

Sintaxis: Cada entidad 1 $\left\{ \begin{array}{l} \text{Debe} \\ \text{Puede} \end{array} \right\}$ ser Nombre de la relación $\left\{ \begin{array}{l} \text{Uno o más} \\ \text{Uno y sólo uno} \end{array} \right\}$ Entidad 2.

- Atributo. Identifica el significado de un dato, que forma parte del sistema.

- Entidad. Es un tipo de objeto definido en base a la agregación de una serie de atributos.¹²

La estructura lógica general de una base de datos se puede expresar gráficamente mediante un *diagrama E-R*. La simplicidad relativa y la claridad pictórica de esta técnica de diagrama puede ser en gran parte la causa del uso del modelo E-R. Tal diagrama consta de los siguientes componentes principales:

- Rectángulos. Representan conjuntos de entidades. 
- Elipses. Representan atributos. 
- Rombos. Representan relaciones. 
- Líneas. Unen atributos a conjuntos de entidades y conjuntos de entidades a conjuntos de relaciones.¹³   debe  puede
- Relación. Debe y puede.

El conjunto de entidades con que se trabaje están relacionadas entre sí, esta relación está determinada por la *correspondencia de cardinalidades*, o razón de cardinalidad, éstas expresan el número de entidades a las que otra entidad puede estar asociada vía un conjunto de relaciones.

La correspondencia de cardinalidades es la más útil describiendo conjuntos de relaciones binarias, aunque ocasionalmente contribuye a la descripción de conjuntos de relaciones que implican más de dos conjuntos de entidades.

Para un conjunto de relaciones binarias R entre los conjuntos de entidades A y B, la correspondencia de cardinalidades debe ser una de las siguientes:

- **Uno a uno**. Una entidad en A se asocia con a lo sumo una entidad en B, y una entidad en B se asocia con a lo sumo una entidad en A.



- **Uno a varios**. Una entidad en A se asocia con cualquier número de entidades en B. Una entidad en B, sin embargo, se puede asociar con a lo sumo una entidad en A.

- **Varios a uno**. Una entidad en A se asocia con a lo sumo una entidad en B. Una entidad en B, sin embargo, se puede asociar con cualquier número de entidades en A.



¹² Diaz, Pablo. ACCESS 2000. Prentice Hall 1999. Pág. 373

¹³ Silberschatz, Abraham. "Fundamentos de Bases de Datos". México. McGraw-Hill. 1998.

- **Varios a varios.** Una entidad en A se asocia con cualquier número de entidades en B, y una entidad en B se asocia con cualquier número de entidades en A.



NORMALIZACIÓN

El proceso de cristalización de las entidades y sus relaciones en formatos de una tabla usando conceptos relacionales se llama **normalización** y a la teoría en la que se basan se le denomina *Teoría de normalización de relaciones*.

Se dice que una relación está en una determinada forma normal si satisface un cierto conjunto específico de restricciones impuestas por la regla de normalización correspondiente.

La aplicación de una regla de normalización es una operación que toma una relación como argumento de entrada y da como resultado dos o más relaciones, y:

- La relación objeto de la aplicación de la regla es desestimada en el nuevo esquema relacional considerado.
- No se introducen nuevos atributos en el esquema relacional resultante de la normalización.
- Los atributos de la relación objeto de la normalización pasan a formar parte de la intención de una o más de las relaciones resultantes.
- En la aplicación de la regla de normalización se ha debido eliminar, al menos, una dependencia existente entre los atributos de la relación objeto de la normalización.

De esta forma, la sucesiva aplicación de las reglas de normalización va a dar lugar a la generación de un número mayor de relaciones que formen parte del esquema relacional y, desde un punto de vista sólo lógico, una redundancia de los atributos considerados en el esquema.

La aplicación sucesiva de las reglas de normalización restringe, por tanto, el número de relaciones que las satisfacen.

Primera forma normal "No hay grupos repetidos."

El primer paso de la normalización consiste en transformar los campos de datos a una tabla de dos dimensiones. Lo que se requiere normalmente en este paso es la eliminación de ocurrencias repetidas de campos de datos, de tal manera que se obtenga un archivo fijo.

Segunda forma normal "Ningún atributo de una no-clave depende de una porción de la clave primaria."

El segundo paso de la normalización es establecer las claves y relaciones con los campos de datos. En la primera forma normalizada, el renglón entero de la tabla (tupla) depende de todos los campos de claves. En la segunda forma normalizada, se hace un intento de establecer los campos de datos que están relacionados con alguna parte de la clave completa. Si los campos de datos sólo dependen de una parte de la clave, la clave y los campos conectados a la clave parcial son

susceptibles de separarse en registros independientes. La división de la primera tabla normalizada, en una serie de tablas en las que cada campo sólo depende de la clave completa se llama segunda forma normalizada.

Tercera forma normal "Ningún atributo depende de otros atributos de una columna no - clave."

El tercer paso consiste en separar los campos de las segundas relaciones normales, que aunque dependan sólo de una clave, deben tener una existencia independiente en la base de datos. Esto se hace de forma tal que la información sobre estos campos pueda introducirse separadamente a partir de las relaciones en las que se encuentra implicada.

En cada modelo de datos uno o más campos de datos se agrupan para representar entidades y sus relaciones. En los agrupamientos de los campos de datos pueden darse tres tipos generales de problemas, y la eliminación de cada uno de éstos da pie a las tres formas normalizadas de relaciones.

Ejemplos¹⁴:

1FN (Primera Forma Normal)	2FN (Segunda Forma Normal)																																				
<p>LIBROS {Código, Título, Autor}</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>CÓDIGO</th> <th>TÍTULO</th> <th>AUTOR</th> </tr> </thead> <tbody> <tr> <td>2154989</td> <td>Modelos de datos</td> <td>Tsichirtzis Lochovsky</td> </tr> <tr> <td>87654353</td> <td>Una guía para DB2</td> <td>Date</td> </tr> <tr> <td>65465465</td> <td>Bases de datos</td> <td>Gardarin Valduriez</td> </tr> </tbody> </table> <p>Esta tabla no está en 1FN. Hay grupos repetidos.</p> <p>Pasamos esta tabla a 1FN, repetimos los atributos para cada uno de los valores del grupo repetido.</p> <p>LIBROS {Código, Título, Autor}</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>CÓDIGO</th> <th>TÍTULO</th> <th>AUTOR</th> </tr> </thead> <tbody> <tr> <td>2154989</td> <td>Modelos de datos</td> <td>Tsichirtzis</td> </tr> <tr> <td>2154989</td> <td>Modelos de datos</td> <td>Lochovsky</td> </tr> <tr> <td>87654353</td> <td>Una guía para DB2</td> <td>Date</td> </tr> <tr> <td>65465465</td> <td>Bases de datos</td> <td>Gardarin</td> </tr> <tr> <td>65465465</td> <td>Bases de datos</td> <td>Valduriez</td> </tr> </tbody> </table> <p>Esta tabla está en 1FN.</p>	CÓDIGO	TÍTULO	AUTOR	2154989	Modelos de datos	Tsichirtzis Lochovsky	87654353	Una guía para DB2	Date	65465465	Bases de datos	Gardarin Valduriez	CÓDIGO	TÍTULO	AUTOR	2154989	Modelos de datos	Tsichirtzis	2154989	Modelos de datos	Lochovsky	87654353	Una guía para DB2	Date	65465465	Bases de datos	Gardarin	65465465	Bases de datos	Valduriez	<p>Si tenemos la relación:</p> <p>PRESTAMOS{Cód_libro, Num_socio, Editorial}, en la que existe la siguiente dependencia:</p> <p style="text-align: center;"><i>Cód_libro</i> —————→ <i>Editorial</i></p> <p>y cuya clave está constituida por el descriptor <i>Cód_libro</i>, <i>Num_socio</i>, esta relación no se encuentra en 2FN al venir la editorial determinada sólo por el código del libro y no por la clave completa. Para que esté en 2FN, descomponemos la relación y la convertimos en las siguientes relaciones:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>PRESTA1{Cód_libro, Num_socio}</td> <td>y así desaparecen las</td> </tr> <tr> <td>PRESTA2{Cód_libro, Editorial}</td> <td>redundancias e inconsistencias</td> </tr> </table> <p style="text-align: center;">3FN (Tercera Forma Normal)</p> <p>Sea, por ejemplo, la relación:</p> <p>SOCIO{Num_socio, Ciudad, País}, que presenta las siguientes dependencias:</p> <p style="text-align: center;"><i>Num_socio</i> —————→ <i>Ciudad</i></p> <p style="text-align: center;"><i>Ciudad</i> —————→ <i>País</i></p> <p>y cuya clave es, evidentemente, <i>Num_socio</i>. Dicha relación no se encuentra en 3FN, al depender <i>País</i> transitivamente de la clave a través de <i>Ciudad</i>. Para que esté en 3FN, descomponemos la relación y la convertimos en las siguientes relaciones:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>SOCIO1{Num_socio, Ciudad}</td> </tr> <tr> <td>Ciudad{Ciudad, País}</td> </tr> </table>	PRESTA1 {Cód_libro, Num_socio}	y así desaparecen las	PRESTA2 {Cód_libro, Editorial}	redundancias e inconsistencias	SOCIO1 {Num_socio, Ciudad}	Ciudad {Ciudad, País}
CÓDIGO	TÍTULO	AUTOR																																			
2154989	Modelos de datos	Tsichirtzis Lochovsky																																			
87654353	Una guía para DB2	Date																																			
65465465	Bases de datos	Gardarin Valduriez																																			
CÓDIGO	TÍTULO	AUTOR																																			
2154989	Modelos de datos	Tsichirtzis																																			
2154989	Modelos de datos	Lochovsky																																			
87654353	Una guía para DB2	Date																																			
65465465	Bases de datos	Gardarin																																			
65465465	Bases de datos	Valduriez																																			
PRESTA1 {Cód_libro, Num_socio}	y así desaparecen las																																				
PRESTA2 {Cód_libro, Editorial}	redundancias e inconsistencias																																				
SOCIO1 {Num_socio, Ciudad}																																					
Ciudad {Ciudad, País}																																					

¹⁴ Miguel Castaño de, Adoración. "Fundamentos y modelos de bases de datos.", España 1997. Págs: 285-288.

CAPÍTULO 4

SELECCIÓN DEL LENGUAJE DE PROGRAMACIÓN

En este capítulo analizaremos algunos lenguajes de programación que permiten crear aplicaciones para realizar consultas a bases de datos de la forma más adecuada. Así, requerimos de un lenguaje de programación que nos garantice un tiempo de vida adecuado y que nos permita llevar a cabo las actualizaciones y cambios necesarios sin causar alguna alteración en la información o en la estructura de la base de datos.

Los lenguajes analizados en función de las características, ventajas y/o desventajas de cada uno serán los siguientes:

VISUAL BASIC

Es un lenguaje de programación que proporciona un juego completo de herramientas que facilitan el desarrollo de aplicaciones.

La palabra *Visual* hace referencia al método que se utiliza para crear la interfaz gráfica de usuario (GUI). En lugar de escribir numerosas líneas de código para describir la apariencia y la ubicación de los elementos que componen la interfaz gráfica, simplemente se agregan los objetos prefabricados que este lenguaje proporciona y se colocan en la pantalla. La palabra *Basic* hace referencia al lenguaje BASIC (Beginners All-Purpose Symbolic Instruction Code).

Sin embargo Visual Basic ha evolucionado a partir del lenguaje BASIC original y ahora contiene centenares de instrucciones, funciones y palabras clave, muchas de las cuales están directamente relacionadas con la interfaz gráfica de Windows¹⁵.

VENTAJAS:

- Permite un diseño de bases de datos Oracle y SQL Server de forma rápida con herramientas visuales para la creación de tablas, relaciones, procedimientos almacenados y funciones.
- Distribuye gratuitamente el motor Microsoft Data Engine (MSDE) como parte de las aplicaciones obteniendo así total compatibilidad con bases de datos más grandes en SQL Server.
- ADO (ActiveX Data Objects) permite el Acceso Universal a Datos a las fuentes de datos más populares, incluyendo Microsoft SQL Server, Microsoft Access, Microsoft FoxPro, Oracle, mainframes IBM y AS/400, así como conectividad mediante ODBC a la mayoría del resto de bases de datos.
- Permite diseñar arquitecturas eficientes de aplicaciones usando Microsoft Visual Modeler, y generar código automáticamente a partir de los diseños de la aplicación.
- Maximiza la productividad con entornos de desarrollo visuales que permiten el arrastrar y soltar componentes, menús y barras de herramientas personalizables, ventanas de propiedades y capacidad de creación de macros personalizados.
- Todas las herramientas de Visual Studio comparten el mismo aspecto y comportamiento.
- Se escribe código más rápidamente y evita errores con la tecnología IntelliSense que comprueba la sintaxis del código y ofrece ayudas conforme se escribe.

¹⁵ <http://www.qualitrain.com.mx/objeIndirecto/javavsvbasic.htm>

- Utiliza la depuración visual multilenguaje en cliente y servidor, incluyendo establecer puntos de parada con el ratón, examinar variables mediante arrastrar y soltar y visualización de múltiples pilas de llamadas.
- Reduce los costos de distribución con asistentes para el empaquetado y distribución automática de aplicaciones y componentes tanto para equipos cliente como para servidores.
- Puede crear aplicaciones auto-reparables usando el Nuevo Visual Studio Installer.
- Optimiza el rendimiento de las aplicaciones con Visual Studio Analyzer, una herramienta gráfica para analizar el rendimiento de aplicaciones distribuidas.
- Organiza de forma efectiva el equipo de desarrollo y beneficia el uso de tecnologías clave para diseño, codificación y gestión de componentes en equipo.

DESVENTAJAS:

- Se debe pagar la licencia por cada equipo que utilice el lenguaje.
- No se puede hacer una consulta en tiempo real y se tendría la necesidad de realizar importaciones y exportaciones constantemente.
- Se tendría que hacer una descarga manual de la información en cada una de las bases de datos a comunicar, con ello se invertiría tiempo de forma innecesaria.

JAVA

Java es un lenguaje de programación orientado a objetos. Moldeado en base a C++, se diseñó para ser pequeño, sencillo y portátil a través de plataforma y sistemas operativos, tanto a nivel de código fuente como en binario.

VENTAJAS

- *Simple*. Elimina la complejidad de los lenguajes como "C" y da paso al contexto de los lenguajes modernos orientados a objetos. Orientado a Objetos. La filosofía de programación orientada a objetos es diferente a la programación convencional.
- *Familiar*. Como la mayoría de los programadores están acostumbrados a programar en C o en C++, la sintaxis de Java es muy similar al de éstos.
- *Robusto*. El sistema de Java maneja la memoria de la computadora. No se tiene que preocupar por apuntadores, memoria que no se esté utilizando, etc. Java realiza todo esto sin necesidad de que el usuario se lo indique.
- *Seguro*. El sistema de Java tiene ciertas políticas que evitan se puedan codificar virus con este lenguaje. Existen muchas restricciones, especialmente para los applets, que limitan lo que se puede y no puede hacer con los recursos críticos de una computadora.
- *Portable*. Como el código compilado de Java (conocido como bytecode) es interpretado, un programa compilado de Java puede ser utilizado por cualquier computadora que tenga implementado el interprete de Java.
- *Independiente a la arquitectura*. Al compilar un programa en Java, el código resultante es un tipo de código binario conocido como bytecode. Este código es interpretado por diferentes computadoras de igual manera, solamente hay que implementar un intérprete para cada plataforma. De esa manera Java logra ser un lenguaje que no depende de una arquitectura computacional definida.
- *Dinámico*. Java no requiere que se compilen todas las clases de un programa para que este funcione. Si se realiza una modificación a una clase Java se encarga de realizar un Dynamic Binding o un Dynamic Loading para encontrar las clases¹⁶.

¹⁶ <http://www.vanco.es/ContentManager/Document.asp?CombinedId=5D306D504>

DESVENTAJAS

- Hay diferentes tipos de soporte técnico para la misma herramienta, por lo que el análisis de la mejor opción se dificulta.
- Para manejo a bajo nivel deben usarse métodos nativos, lo que limita la portabilidad.
- El diseño de interfaces gráficas con awt y swing no es simple.
- Existen herramientas como el JBuilder que permiten generar interfaces gráficas de manera sencilla, pero tienen un costo adicional.
- Puede ser que no haya JDBC para bases de datos poco comerciales.
- Algunas herramientas tienen un costo adicional.

PHP

PHP son las siglas "Personal Home Page". Es un lenguaje de programación pensado en la web de forma que es ideal para la creación de páginas dinámicas. PHP es la versión libre del sistema equivalente de Microsoft ASP.

PHP es un lenguaje encapsulado dentro de los documentos html (aunque los CGI hechos con PHP terminan en extensión .php y no .html). De forma que se pueden introducir instrucciones php dentro de las páginas. Gracias a esto el diseñador gráfico de la web puede trabajar de forma independiente al programador. PHP es interpretado por el servidor (apache) generando un HTML con el resultado de sustituir las secuencias de instrucciones PHP por su salida.

Por lo tanto una web dinámica con PHP contiene una serie de documentos php que el servidor apache interpreta proporcionando al cliente documentos html con el resultado de las ordenes php.

Dispone de múltiples herramientas que permiten acceder a bases de datos de forma sencilla, por lo que es ideal para crear aplicaciones para Internet. Es multiplataforma, funciona tanto para Unix (con Apache) como para Windows (con Microsoft Internet Information Server) de forma que el código que se haya creado para una de ellas no tiene por qué modificarse al pasar a la otra.

La sintaxis que utiliza la toma de otros lenguajes muy extendidos como C y Perl, por lo que se está familiarizado con éstos, PHP será fácilmente utilizado.

VENTAJAS

- Muy sencillo de aprender.
- Similar en sintaxis a C y a Perl.
- Soporta en cierta medida la orientación a objetos, clases y herencia.
- El análisis léxico para recoger las variables que se pasan en la dirección lo hace PHP de forma automática. Librándose el usuario de tener que separar las variables y sus valores.
- Se puede incrustar código PHP con etiquetas HTML.
- Excelente soporte de acceso a base de datos.
- La comprobación de que los parámetros son válidos se hace en el servidor y no en el cliente de forma que se puede evitar el revisar que no se reciban solicitudes adulteradas. Además PHP viene equipado con un conjunto de funciones de seguridad que previenen la inserción de órdenes dentro de una solicitud de datos.
- Se puede hacer todo lo que se pueda transmitir por vía http.

DEVENTAJAS

- Todo el trabajo lo realiza el servidor y no delega al cliente. Por lo tanto puede ser más ineficiente a medida que las solicitudes aumenten de número.
- La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP.
- La orientación a objetos es aún muy deficiente para aplicaciones grandes.

POWERBUILDER

PowerBuilder es un lenguaje de programación orientado a eventos que permite desarrollar diversas aplicaciones en entorno Windows de forma sencilla y rápida.

Puede utilizarse tanto en el desarrollo de aplicaciones para el uso de bases de datos como en el de sistemas de gestión de bases de datos. Las aplicaciones desarrolladas en PowerBuilder pueden ser muy sencillas o complejas, pero todas tienen la apariencia de windows y se pueden manejar fácilmente con el mouse. PowerBuilder es un desarrollador de aplicaciones para el ambiente Windows, como tal utiliza las características de este ambiente. Primero debemos entender que el ambiente Windows permite estar bajo el concepto "Lo que Ud. ve es lo que puede realizar", bajo este contexto la presentación cobra mucha importancia.

Este objeto, llamado *DataWindow*, incorpora en su interior, el mecanismo necesario para efectuar el acceso a datos, ya sea vía ODBC, o a través de una interfaz nativa. Así mismo, proporciona un mecanismo de selección de los datos, absolutamente intuitivo, que junto con la capacidad de heredar características de un objeto, por parte de otro, constituye una fuente de desarrollo requerido, para la construcción de cierto tipo de aplicaciones. Además se proporciona por parte de este tipo de objeto, la opción de generar su salida, directamente en un archivo con formato HTML, que puede ser interpretado directamente por un navegador de Internet.

VENTAJAS

- Soporta una gran variedad de sistemas de gestión de base de datos, tales como: Sybase, Informix, Oracle, Access, entre otras.
- Tiene capacidad de acceder a información de múltiples bases de datos y mostrar esa información en una única ventana.
- Se trabaja en ambiente cliente - servidor.
- Posee un objeto inteligente llamado *Datawindow* que realiza directamente la interfaz con la base de datos, sin requerir que el programador conozca SQL.
- Capacidad de utilizar sentencias SQL combinadas en el código.
- Se puede trabajar en múltiples plataformas, ya que soporta diferentes sistemas operativos y posee drivers nativos para las bases de datos más comerciales.
- PowerBuilder proporciona los medios necesarios para el desarrollo de aplicaciones de potentes sistemas de tratamiento de datos, en su mayor parte remotos, sistemas de gestión de datos robustos y, como la mayor parte de ellos, soportado a través de *Intranet/Internet*
- Se puede generar fácilmente código reutilizable, que revierta en una menor inversión en el tiempo de desarrollo, así como la fiabilidad y robustez de las aplicaciones construidas sobre dicho código.

La creación de aplicaciones es sencilla.

- Se realiza poca programación.
- Permite usar ventanas, botones y todas las herramientas que presenta windows facilitando su manejo.
- Para construir la aplicación se utilizan painters, allí se definen las propiedades de los objetos y se agregan los controles.
- Se puede trabajar con múltiples ventanas.

DESVENTAJAS

- Ambiente de programación jerárquico, un poco diferente al normal. Cambiar de dueño es una incertidumbre.
- Quizás es la más difundida herramienta del mercado, cuenta con un gran soporte de terceros pero podría resultar una excelente opción si incorporara a ésta su DBMS.

El siguiente cuadro resume las características principales de los lenguajes de programación.

Lenguaje	Ventajas	Desventajas
Visual Basic	<ul style="list-style-type: none"> ➤ Permite la comunicación entre diferentes Bases de Datos. ➤ Ambiente Cliente/Servidor. ➤ Creación de aplicaciones de servidor para Internet. 	<ul style="list-style-type: none"> ➤ Requiere el uso de Access. ➤ No se realizan consultas en tiempo real.
Java	<ul style="list-style-type: none"> ➤ Objetos gráficos independientes a la plataforma. ➤ Manejo de Bases de Datos transparente y simple. 	<ul style="list-style-type: none"> ➤ No se cuenta con soporte técnico confiable debido a la variedad del mismo. ➤ Portabilidad limitada.
PHP	<ul style="list-style-type: none"> ➤ Soporte de acceso a Bases de Datos. ➤ Funciones de Seguridad. 	<ul style="list-style-type: none"> ➤ Ineficiente a medida que aumenta la cantidad de solicitudes. ➤ Orientación a objetos deficiente para aplicaciones grandes.
Power Builder	<ul style="list-style-type: none"> ➤ Variedad de Sistemas de Gestión de Bases de Datos. ➤ Sistemas de Gestión de Datos soportados a través de Internet/Intranet. ➤ Encripta la información al enviarla a su destino. 	<ul style="list-style-type: none"> ➤ Ambiente de programación jerárquico. ➤ Costo de Licencia.

ELECCIÓN DEL LENGUAJE DE PROGRAMACIÓN

Una vez que se analizaron los lenguajes de programación que podrían servirnos para el desarrollo de la propuesta llegamos a la siguiente conclusión.

Se decidió que el lenguaje de programación adecuado para crear la propuesta es PowerBuilder. La principal característica que nos llevó a decidir este lenguaje de programación es que las ventajas descritas de éste, permiten el acceso y la administración de la información en una base de datos de

manera sencilla, la programación de interfaces que viene realizando la USECAD en este lenguaje ha tenido resultados exitosos manejando grandes volúmenes de datos, permitiéndonos estimar resultados similares con un manejo de datos en proporciones menores.

Resumiendo, utilizar dicho lenguaje nos permitirá realizar consultas de forma más simple a la información contenida en la base de la USECAD. Otro aspecto decisivo fue que trabaja adecuadamente con el SGBD que utiliza la USECAD (Sybase) y además, es con éste con el que administra su base de datos.

Con este lenguaje podemos hacer consultas a la base de datos de USECAD utilizando una arquitectura Cliente/Servidor de 2 capas¹⁷, arquitectura que es utilizada por USECAD.

¹⁷ La definición de la arquitectura cliente/servidor se realiza en el Capítulo 5 "Redes de computadoras"

CAPÍTULO 5

REDES DE COMPUTADORAS

En este capítulo vamos a realizar un análisis de los diferentes tipos de redes de computadoras, sus ventajas y desventajas, así como las topologías existentes con el fin de establecer el que se ajuste a nuestras necesidades en cuanto a seguridad de la información y tiempo de envío de la misma, o en su defecto, verificar que el tipo de red y topología que utiliza la USECAD garantizan que al realizar el envío de información desde el Departamento de Ingeniería Mecánica hasta USECAD no se tendrá pérdida o alteración alguna en la misma.

RED DE COMPUTADORAS.

Es una colección interconectada de computadoras y dispositivos independientes que ofrece intercambio interno entre medios de voz, datos y video¹⁸.

El enlace entre las máquinas se realiza primero a través de un medio físico y posteriormente a través de un medio llamado protocolo.

Ventajas de las redes de computadoras

1. Posibilidad de compartir periféricos costosos, como impresoras láser, módem, fax, etc.
2. Posibilidad de compartir grandes cantidades de información a través de distintos programas, bases de datos, etc., de manera que sea más fácil su uso y actualización.
3. Reduce e incluso elimina la duplicidad de trabajos.
4. Permite utilizar el correo electrónico para enviar y recibir mensajes de diferentes usuarios de la misma red e incluso de redes diferentes.
5. Reemplaza o complementa mini computadoras de forma eficiente y con un costo bastante más reducido.
6. Establece enlaces con mainframes. De esta forma, una computadora de gran potencia actúa como servidor, haciendo que los recursos disponibles estén accesibles para cada una de las computadoras personales conectadas.
7. Permitir mejorar la seguridad y control de la información que se utiliza, admitiendo la entrada de determinados usuarios, accediendo únicamente a cierta información o impidiendo la modificación de diversos datos.

CLASIFICACIÓN DE LAS REDES.

a) De acuerdo a su alcance geográfico

Una clasificación de los sistemas de información en general puede ser de acuerdo con su tamaño físico y alcance geográfico. En el caso de las redes tenemos las siguientes:

¹⁸ Apuntes de la materia "Redes de Computadoras " impartida por los profesores Ing. Marco Antonio López Vega (del tema de "Redes de Datos") e Ing. Marco Antonio Viguera Villaseñor; Facultad de Ingeniería, Universidad Nacional Autónoma de México. 2002.

LAN (Local Area Network)

Son las redes cuyas comunicaciones están limitadas a un área geográfica de dimensiones moderadas (edificio, oficina), se puede decir que su extensión puede darse entre los 0.1 Km. hasta los 25 Km. En este tipo de redes pueden verse las siguientes características

- Los canales son propios de los usuarios o empresas.
- Los enlaces son líneas de alta velocidad.
- Las estaciones están cerca entre sí.
- Incrementan la eficiencia y productividad de los trabajos de oficinas al poder compartir información.
- Las tasas de error son menores que en las redes WAN.
- La arquitectura permite compartir recursos.
- LANs mucha veces usa una tecnología de transmisión, dada por un simple cable, donde todas las computadoras están conectadas.

MAN (Metropolitan Area Network)

Redes de área metropolitana (Metropolitan Area Network) con dos buses unidireccionales, cada uno de ellos es independiente del otro en cuanto a la transferencia de datos. Es básicamente una gran versión de LAN y usa una tecnología similar. Puede cubrir un grupo de oficinas de una misma corporación o ciudad, esta puede ser pública o privada. El mecanismo para la resolución de conflictos en la transmisión de datos que usan las MANs, es DQDB.

DQDB consiste en dos buses unidireccionales, en los cuales todas las estaciones están conectadas, cada bus tiene una cabecera y un fin. Cuando una computadora quiere transmitir a otra, si esta está ubicada a la izquierda usa el bus de arriba, caso contrario el de abajo.

WAN (Wide Area Network)

Red de área amplia (Wide Area Network). El alcance es una gran área geográfica, como por ejemplo: una ciudad o un continente. Está formada por una vasta cantidad de computadoras interconectadas (llamadas hosts), por medio de subredes de comunicación o subredes pequeñas, con el fin de ejecutar aplicaciones, programas, etc.

Una subred está formada por dos componentes:

1. **Líneas de transmisión:** quienes son las encargadas de llevar los bits entre los hosts.
2. **Elementos interruptores (routers):** son computadoras especializadas usadas por dos o más líneas de transmisión. Para que un paquete llegue de un router a otro, generalmente debe pasar por routers intermedios, cada uno de estos lo recibe por una línea de entrada, lo almacena y cuando una línea de salida está libre, lo retransmite¹⁹.

¹⁹ <http://www.linti.unlp.edu.ar/trabajos/tesisDeGrado/tutorial/redes/tipredes.htm>

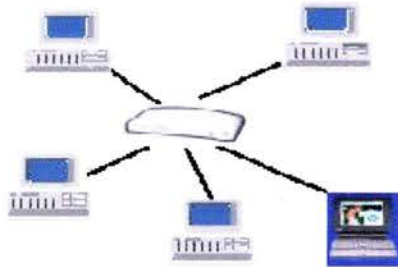
b) De acuerdo a la topología

En el diseño de una red es muy importante considerar una topología la cual es la forma en que una red se constituye físicamente.

ESTRELLA

Uno de los tipos más antiguos de topologías de redes es la estrella, en donde las computadoras se conectan a un dispositivo central de conexiones (conocido como concentrador o hub), el cual controla el flujo de datos.

Esta es una buena configuración para actividades comerciales, que tienen grandes cantidades de información que sufre rápidos cambios, como los Bancos y las oficinas de reserva de pasajes aéreos.

EstrellaCARACTERÍSTICAS

- Servidor centralizado.
- El nodo central es el responsable de la comunicación entre nodos.
- Comunicaciones de tipo bidireccionales.

VENTAJAS

- Simple para interconectar.
- Si falla un nodo en este esquema de red no afecta la funcionalidad de la misma.
- Es una de las topologías más rápidas en situaciones de tráfico pesado (*por el criterio de enrutamiento que sigue el servidor*).
- Requiere de software mucho más simple para los dispositivos individuales.
- Provee control cercano a los datos.
- Cada PC puede ver todos los datos.

- Es segura para realizar el envío de información.
- Los daños son fáciles de detectar y corregir.
- Utiliza el Protocolo TCP/IP.

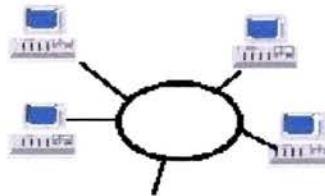
DESVENTAJAS

- Si falla el nodo principal, falla toda la red. *(Sería necesario tener otro sistema de computadora disponible como backup para mantener todo funcionando mientras se hacen las reparaciones).*
- Requiere de mayor medio físico para la interconexión de dispositivos. *(se utiliza mucho cable)²⁰.*
- Es una red costosa por la cantidad de cableado que debe utilizarse.
- Su instalación es compleja.
- La velocidad de transmisión es un poco lenta.

ANILLO

Una topología de anillo consta de varios nodos unidos formando un círculo lógico. Los mensajes se mueven de nodo a nodo en una sola dirección. Algunas redes de anillo pueden enviar mensajes en forma bidireccional, no obstante, sólo son capaces de enviar mensajes en una dirección cada vez. La topología de anillo permite verificar si se ha recibido un mensaje.

Anillo



VENTAJAS

- Los cuellos de botellas son muy pocos frecuentes.
- Requiere menos cableado y por consiguiente es más económica.
- Combinación perfecta de protocolos.
- Mejor ancho de banda.

²⁰ <http://iio.ens.uabc.mx/jmilanez/escolar/redes/01110100.html>

- Trabaja en tiempo real todos los medios de transmisión.

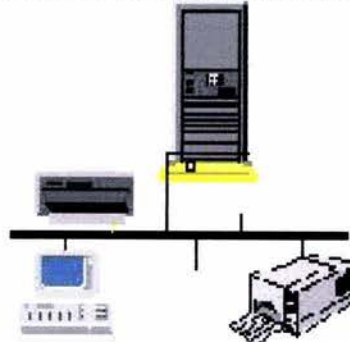
DESVENTAJAS

- Al existir un solo canal de comunicación entre las estaciones de la red, si falla el canal o una estación, las restantes quedan incomunicadas.
- Algunos fabricantes resuelven este problema poniendo un canal alternativo para casos de fallos, si uno de los canales es viable la red está activa, o usando algoritmos para aislar las componentes defectuosas.
- Es muy compleja su administración, ya que hay que definir una estación para que controle el token.
- No es muy Segura para transmitir información.
- Existe un mecanismo para la resolución de conflictos en la transmisión de datos²¹.

BUS LINEAL

En una topología de bus, cada computadora está conectada a un segmento común de cable de red. Éste se coloca como un bus lineal, es decir, un cable largo que va de un extremo a otro de la red, siempre y cuando sea continuo y al cual se conecta cada nodo.

Bus lineal



VENTAJAS

- Todos las computadoras están conectados entre sí.
- Rápida transmisión de la información.
- Fácil de instalar.
- Utiliza protocolos fácilmente entendibles.
- Bajos Costos.

²¹ <http://www.monografias.com>

- Comunicación Bidireccional.

DESVENTAJAS

- Todos tienen acceso a la información.
- No hay control del flujo de información.
- No disponibles los medios de transmisión.
- Se manejan protocolos a nivel de capa 1 del modelo OSI.

TRANSMISIÓN DE DATOS EN LAS REDES

La transmisión de datos en las redes puede ser por dos medios:

1. Terrestres: Son limitados y transmiten la señal por un conductor físico. Estos pueden ser:
 - a. Cable eléctrico de pares trenzados.
 - b. Cable coaxial en banda base para velocidades de transmisión de hasta 50 Mbps.
 - c. Cable coaxial en banda ancha para velocidades de transmisión de más de 300 Mbps.
 - d. Cable de fibra óptica para velocidades de transmisión de más de 150 Mps.
2. Aéreos: Son "ilimitados" en cierta forma y transmiten y reciben las señales electromagnéticas por microondas o rayo láser. Éstos pueden ser:
 - a. Ondas de cielo.
 - b. Ondas terrestres.
 - c. Microondas.
 - d. Sistemas satelitales.
 - e. Sistemas geosíncronos.

PROTOCOLOS

Al conjunto de reglas que regulan el flujo o intercambio de información entre los diferentes elementos de un sistema distribuido o de cualquier sistema comunicado, se le denomina protocolo.

Los protocolos son acuerdos, mediante combinaciones de caracteres, que establecen quién debe hacer, qué hacer y cuándo hacerlo.

El protocolo permite fundamentalmente iniciar, mantener y terminar un diálogo entre los elementos del sistema.

OSI y TCP/IP

El modelo OSI establece los lineamientos para que el software y los dispositivos de diferentes fabricantes funcionen juntos. Aunque los fabricantes de hardware y los de software para red son los usuarios principales del modelo OSI, una comprensión general del modelo llega a resultar muy benéfica para el momento en que se expande la red o se conectan redes para formar redes de área amplia (WAN)²². Las capas del modelo OSI son:

- **Capa 7.-** La capa de Aplicación funciona como el acceso a los servicios que proporciona la red, así como de proporcionar al sistema operativo servicios como el de la transferencia de archivos.
- **Capa 6.-** La función de la capa de Presentación es la de proveer una interfaz para realizar la transferencia de datos que sea idéntica de la tecnología para representarlos. Los datos pueden representarse en varias formas, lo que define como usar los datos y como mostrarlos es la arquitectura del sistema, así que la capa de presentación se encarga de esto.
- **Capa 5.-** La capa de sesión tiene la responsabilidad de asegurar la entrega correcta de la información. Esta capa tiene que revisar que la información que recibe este correcta; para esto, la capa de sesión debe realizar algunas funciones:
 - * La detección y corrección de errores.
 - * El controlar los diálogos entre dos entidades que se estén comunicando y definir los mecanismos para hacer las llamadas a procedimientos remotos (*Remote Procedure Control - RPC*).

Hasta aquí, las tres primeras capas son denominadas "Capas de host" o las capas más dependientes de la computadora o del anfitrión local (o *incluso dentro del mismo programa*). Las últimas tres capas están orientadas hacia la comunicación (*hacia la red*).

El TCP ejecuta funciones importantes en la capa de sesión.

- **Capa 4.-** La capa de transporte vincula las capas de host con las capas orientadas a la red; permite la cohesión entre el host y la red, su función es la de asegurar una entrega confiable de la información a través de la red. Los estándares que pertenecen a la capa de transporte incluyen el protocolo de transporte (TP) de la Organización Internacional de Estándares (ISO) y el protocolo de intercambio de paquetes en secuencia (SPX) de Novell. Otros estándares que ejecutan funciones importantes en la capa de transporte incluyen el protocolo de control de transmisión (TCP) del Departamento de la Defensa, que es parte del TCP/IP, y el NCP de Novell.
- **Capa 3.-** Incluye dos cosas fundamentales: la capa de Red se encarga de determinar las rutas adecuadas para llevar la información de un lado a otro (proporciona el enrutamiento); además, su funcionalidad es la de proporcionar una interfaz para que la transferencia de datos sea idéntica de la tecnología del enlace de datos. Los estándares que se refieren a la capa de red incluyen el protocolo de intercambio de

²² <http://iio.ens.uabc.mx/jmilanez/escolar/redes/03020000.html>

paquetes entre redes (IPX) de Novell, el protocolo de Internet (IP) y el protocolo de entrega de datagramas (DDP) de Apple. El IP es parte del estándar de protocolo TCP/IP, generado por el Departamento de la Defensa de Estados Unidos y utilizado en Internet. El DDP fue diseñado para computadoras Apple, como la Macintosh. Los enrutadores operan en la capa de red.

- **Capa 2.-** La función de la capa dos es la de asegurar la transferencia de datos libres de error entre nodos adyacentes (sincronización a nivel de datos), además establece el control de acceso al medio. La capa de enlace de datos está dividida en dos subcapas: el control de acceso al medio (MAC) y el control de enlace lógico (LLC). Los puentes (*bridges*) operan en la capa MAC.

- * Control de enlace lógico.
- * IEEE 802.2 (enlace lógico).
- * Punto a Punto (PPP).

MAC.

- * IEEE 802.3 - CSMA/CD.
- * IEEE 802.5 - Token Ring.
- * ANSI FDDI - Token Ring (fibra).

- **Capa 1.-** Define las características físicas del medio de transmisión; de tipo mecánico, eléctrico y óptico (esto es, el tipo de medio a utilizar, el tamaño o forma de los conectores, el grosor del cable, el tipo de cable, el tipo de aislante, el voltaje de la interfase, la impedancia - resistencia - nominal, etc.), además esta la señalización de la interfase (es decir, el como representar la información como un 0 y 1, por ejemplo, un 0 puede representarse como una señal entre 0 y 5 volts, y un 1 en una señal de entre 1 y -5 volts, por ejemplo).

La capa física también maneja los tipos y las especificaciones de cables, incluyendo los cables Ethernet 802.3 del IEEE (Thick Ethernet - Ethernet denso o estándar -, Thin Ethernet - Ethernet estrecho o delgado - y UTP), el estándar de interfaz de datos distribuidos por fibra óptica (FDDI) del Instituto Nacional de Estándares Americanos (ANSI) para el cable de fibra óptica y muchos otros.

MODELO OSI			
Capa		Protocolos asociados	Mecanismos de Conectividad
7	APLICACIÓN	NFS, X.400, X.500, Shell, Redirector	
6	PRESENTACIÓN	RFS, SMB, NCP, NFS	
5	SESIÓN	TCP, IPX, NetBIOS, FTP/Telnet, SMTP, TFTP, RPC, SNMP	
4	TRANSPORTE	TCP, UDP, SPX/IPX	Gateway (Enrutador)
3	CAPA DE RED	IP, IPX, ICMP, X.25	Enrutador
2B	ENLACE DE DATOS LLC	IEEE 802.2, ODI, LABP, NDIS, Drivers	
2A	ENLACE DE DATOS MAC	IEEE 802.3, IEEE 802.5, CSMA/CD, Token	Switch, Puente (Bridge)
1	FÍSICA	IEEE 802.3, IEEE 802.4, IEEE 802.5, RS-232, RS-449, V.35, Topologías	Repetidor, Transceiver, MAU, Hub, NIC, Cableado

SEGURIDAD EN LAS REDES DE COMPUTADORAS

La pérdida de datos insustituibles es una amenaza real para cualquier empresa o centro educativo que conecta su red con el mundo exterior. Por otra parte, el acceso remoto y la conexión a Internet permiten mejorar la comunicación a un nivel sin precedente. No obstante, estas mismas oportunidades exponen las redes locales a sufrir ataques de "hackers", así como al uso inadecuado por parte de sus propios empleados.

Para comprender el nivel de seguridad que una red requiere, es necesario considerar varios factores. En primer lugar, se debe determinar cuál es el valor de los datos. Cuando se evalúan, se deben tomar en cuenta los riesgos, tales como la responsabilidad legal, la pérdida en la productividad en el área de trabajo como consecuencia de haber comprometido la red, etc.

Algunas de las medidas principales de seguridad en redes tenemos las siguientes:

FIREWALLS

Un firewall es un conjunto de programas relacionados entre sí, ubicados en un servidor de gateway de red, que aísla los recursos de una red privada frente a los usuarios y a otras redes. Una empresa conectada a Internet instala un firewall para impedir que otros puedan acceder a sus recursos de datos privados y para controlar los recursos externos a los que tienen acceso sus propios usuarios.

Un firewall filtra todos los paquetes de la red y determina si reenviarlos hacia su destino. Un firewall también puede incluir o trabajar con un servidor proxy que realiza peticiones a la red por parte de los usuarios de estaciones de trabajo. Un firewall se instala siempre en un servidor dedicado, separado del resto de la red (o constituye un dispositivo independiente) para que ninguna petición entrante pueda llegar directamente a los recursos privados de la red.

REDES PRIVADAS VIRTUALES

Una red privada virtual (Virtual Private Network, o VPN) es una red de datos privada que utiliza la infraestructura pública de telecomunicaciones, manteniendo la privacidad mediante el uso de un túnel y un procedimiento de seguridad. Una VPN puede compararse con un sistema de líneas alquiladas o en propiedad que sólo puede ser utilizado por una sola empresa. La idea de la VPN es ofrecer a la empresa las mismas funciones por un coste mucho más bajo, al usar la infraestructura pública compartida en lugar de una privada. Con una VPN, es posible tener el mismo nivel de compartición segura de recursos públicos para datos. Las empresas de hoy contemplan el uso de las VPNs tanto para extranets como para intranets.

El uso de una VPN implica la encriptación de los datos antes de enviarlos por la red pública, y su desencriptación en el extremo del destinatario. Un nivel adicional de seguridad encripta no sólo los datos, sino también las direcciones de red de origen y destino.

ANTI-VIRUS

El software anti-virus es un tipo de programa que detecta virus conocidos o potenciales. El mercado de este tipo de programa ha aumentado debido al crecimiento de Internet y el uso cada vez más frecuente de Internet por empresas preocupadas por la protección de sus activos informáticos.

SEGURIDAD DE E-MAIL/CONTENIDO WEB

A medida que se extiende el uso del correo electrónico y las comunicaciones web, también aumenta el número y la complejidad de las potenciales amenazas. Las organizaciones necesitan poder establecer y hacer cumplir una Seguridad de Contenidos basada en políticas para controlar estas amenazas. La Seguridad de Contenidos basada en políticas depende de que una organización establezca una política aceptable de uso de e-mails y de la web, eduque sus empleados sobre dicha política, y haga cumplir la política mediante una solución de software apropiada. En resumen, significa que una organización necesita poder comprender y gestionar qué datos han sido movidos y adónde, y quién ha tenido acceso a ellos.

Es esencial disponer de sofisticadas soluciones de software que puedan aplicar diferentes políticas de Seguridad de Contenidos a diferentes situaciones. Las organizaciones aplican políticas diferentes a cada objeto que atraviesa su gateway de red, y hacen cumplir estas políticas por individuos y por departamentos, para tener cubiertas cuestiones de integridad tanto del negocio como de la red. Algunas de las cuales son:

INTEGRIDAD DEL NEGOCIO

- infracciones de confidencialidad
- daños a la reputación por mal uso del e-mail y la web
- responsabilidad legal
- pérdidas de productividad
- pornografía
- robo de datos vía e-mail o cuentas de correo tipo 'hotmail'
- degradación de servicios

INTEGRIDAD DE LA RED

- corrupción e infección de datos por virus que acceden vía e-mail
- pérdida de servicio de la red por ataques de spam y spoof
- congestión de la red debido al mal uso de los sistemas

AUTENTICACIÓN DE DOS FACTORES

La autenticación es el proceso mediante el cual se determina si alguien o algo es realmente quien dice o lo que dice ser. En las redes de ordenadores privadas y públicas (incluyendo Internet), la autenticación se lleva a cabo habitualmente mediante el uso de contraseñas de conexión. El conocimiento de la contraseña supuestamente garantiza que el usuario es auténtico. Cada usuario se registra inicialmente (o es registrado por otra persona) usando una contraseña asignada o declarada. En cada uso posterior, el usuario debe conocer y utilizar la contraseña anteriormente establecida. La debilidad de este sistema para transacciones significativas (por ejemplo, el intercambio de dinero) es que las contraseñas a menudo pueden ser robadas, accidentalmente divulgadas u olvidadas.

Por esta razón, el comercio electrónico y muchas otras transacciones requieren un proceso de autenticación más estricto. Un proceso que utiliza un sistema de autenticación basado en una combinación cambiante de contraseña y código basado en el tiempo. La Autenticación de dos factores se basa en algo que usted conoce (una contraseña o PIN) y algo que usted posee (un autenticador) – lo cual ofrece un nivel mucho más fiable de autenticación de usuarios que las contraseñas reutilizables.

DETECCIÓN DE INTRUSOS

La detección de intrusos (DI) es un tipo de sistema de gestión de seguridad para ordenadores y redes. Un sistema DI reúne y analiza información de varias áreas dentro de un ordenador o una red para identificar posibles violaciones de seguridad, que incluyen tanto intrusos (ataques desde fuera de la organización) como mal uso (ataques desde dentro de la organización). La DI utiliza evaluación de vulnerabilidad (a veces llamada exploración), una tecnología desarrollada para poder evaluar la seguridad de un sistema o una red de ordenadores. Las funciones de detección de intrusos incluyen:

- Monitorización y análisis de actividades tanto del usuario como del sistema
- Análisis de la configuración y la vulnerabilidad del sistema
- Evaluación de la integridad del sistema y de los archivos
- Capacidad de reconocer patrones de ataque típicos
- Análisis de patrones anormales de actividad
- Seguimiento de violaciones de políticas por parte de los usuarios

Los sistemas DI se están desarrollando como respuesta al número cada vez mayor de ataques a importantes sitios y redes. Mantener la seguridad es cada vez más difícil, porque las posibles tecnologías de ataque vienen siendo cada vez más sofisticadas; y al mismo tiempo, incluso el atacante novato requiere menos conocimientos técnicos al estar fácilmente disponibles en la web los métodos usados con éxito en el pasado.

Típicamente, un sistema de DI es un proceso de dos etapas. Los procedimientos iniciales están basados en el host, y son considerados como componente pasivo; incluyen: inspección de los archivos de configuración del sistema para detectar valores poco recomendables; inspección de los archivos de contraseñas para detectar contraseñas poco recomendables; e inspección de otras áreas del sistema para detectar violaciones de políticas. Los procedimientos de la segunda etapa están basados en la red, y son considerados como componente activo; se establecen mecanismos para reconstruir métodos de ataque conocidos y registrar las respuestas del sistema.

HTTPS

HTTPS es el protocolo empleado para visualizar sitios Web seguros. A las siglas HTTP sobre las que ya estamos familiarizados se ha añadido una nueva letra. Esta letra o símbolo es 's' se identifica con protocolo seguro de páginas web a través de Internet.. Un sitio Web que disponga de certificado digital ofrece todas las prestaciones de sitio Web seguro generando un complicado algoritmo de información encriptada cuando un usuario envía un formulario contenido en alguna página del sitio Web. Cuando una empresa dispone de un certificado digital puede obligar a que los usuarios le visiten lo hagan de dos formas bien distintas²³.

ARQUITECTURA CLIENTE/SERVIDOR.

En los años 70 era de uso común que todo el procesamiento de las bases de datos se realizara en un gran servidor denominado host, que podía ser una gran máquina o una minicomputadora, quedando solamente las terminales para el ingreso y la visualización de los datos.

Características de un sistema cliente/servidor

Un sistema cliente/servidor es aquel en el que uno o más clientes y uno o más servidores, conjuntamente con un sistema operativo subyacente y un sistema de comunicación entre procesos, forma un sistema compuesto que permite cómputo distribuido, análisis, y presentación de los datos.

Los clientes, a través de la red, pueden realizar consultas al servidor. El servidor tiene el control sobre los datos; sin embargo los clientes pueden tener datos privados que residen en sus computadoras. Las principales características de la arquitectura cliente/servidor son:

- El servidor presenta a todos sus clientes una interfaz única y bien definida.
- El cliente no necesita conocer la lógica del servidor, solo su interfaz externa.
- El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.
- Los cambios en el servidor implican pocos o ningún cambio en el cliente.

²³ <http://www.vanco.es/ContentManager/Document.asp?CombinedId=5D306D504>

Partes de un sistema cliente/servidor

Los principales componentes de un sistema cliente/servidor son:

- El núcleo (back-end o sección posterior). Es el DBMS propiamente (servidor).
- La interfaz (front-end o sección frontal). Aplicaciones que funcionan sobre el DBMS (cliente)

La sección frontal

Las secciones frontales son las diversas aplicaciones ejecutadas dentro del DBMS, tanto las escritas por los usuarios como las "integradas" que son las proporcionadas por el proveedor del DBMS o bien por otros proveedores de programas (aunque para la sección posterior no existe diferencia entre las aplicaciones escritas por los usuarios y las integradas, ya que todas utilizan la misma interfaz con la sección posterior).

Funciones del cliente

- Administrar la interfaz gráfica de usuario (GIU).
- Aceptar datos del usuario.
- Procesar la lógica de la aplicación.
- Generar las solicitudes para la base de datos.
- Transmitir las solicitudes de la base de datos al servidor.
- Recibir los resultados del servidor.
- Dar formato a los resultados.

Como trabaja la sección frontal

Primero, el usuario crea la consulta. Puede ser una consulta creada en el instante o puede ser una consulta programada o almacenada anteriormente. Después la aplicación cliente convierte la consulta al lenguaje usado por el servidor de la base de datos y la envía a través de la red al servidor.

El servidor verifica que el usuario tiene los derechos de seguridad apropiados a la consulta de datos requerida. Si es así, verifica la consulta y envía los datos apropiados de vuelta al cliente. La aplicación cliente recibe la respuesta y le da formato para presentarlo al usuario.

Finalmente, el usuario ve la respuesta en la pantalla y puede manipular los datos, o modificar la consulta y empezar el proceso de nuevo.

La sección posterior

La sección posterior es el DBMS en sí. Permite llevar a cabo todas las funciones básicas de un DBMS: definición de datos, manipulación de datos, seguridad, integridad, etc... En particular, permite establecer todos los aspectos de los niveles externo, conceptual e interno.

Funciones del servidor

- Aceptar las solicitudes de la base de datos de los clientes.
- Procesar dichas solicitudes.
- Dar formato a los resultados y transmitirlos al cliente.
- Llevar a cabo la verificación de integridad.
- Mantener los datos generales de la base de datos.
- Proporcionar control de acceso concurrente.
- Llevar a cabo la recuperación.
- Optimizar el procesamiento de consultas y actualización.

Existen diferentes tipos de Servidores, estos se clasifican de acuerdo al funcionamiento que presentan, entre estos diferentes tipos podemos encontrar:

1. Servidores Iterativos.

- Espera que llegue una consulta de un cliente.
- Procesa la consulta.
- Envía la respuesta al cliente que envió la consulta.
- Vuelve al estado inicial.

El **problema** de este tipo servidor es el segundo punto. Durante el tiempo en el que el servidor está procesando la consulta, ningún otro cliente es servido.

2. Servidores Concurrentes.

- Espera que llegue la consulta de un cliente.
- Cuando le llega una nueva consulta, comienza un nuevo proceso para manejar esta consulta. El nuevo servidor maneja la totalidad de la consulta. Cuando se ha procesado completamente, este nuevo proceso termina.
- Se vuelve al primer paso.

La **ventaja** de estos servidores es que ejecutan un nuevo proceso para manejar cada consulta. Cada cliente tiene su "propio" servidor. Asumiendo que el Sistema Operativo permite la multiprogramación y clientes múltiples.

3. Servidores de Transacciones (también llamados Servidores de Consultas).

- Proporcionan una interfaz a través de la cual los clientes pueden enviar peticiones para realizar una acción que el servidor ejecutará y cuyos resultados se devolverán al cliente.
- Los usuarios deben especificar sus peticiones con SQL o mediante la interfaz de una aplicación utilizando un mecanismo de llamadas a procedimientos remotos (RPC Remote Procedure Call).

4. Servidores de Datos.

- Permiten que los clientes puedan interactuar con los servidores realizando peticiones de lectura o modificación de datos en unidades tales como archivos o páginas.
- Proporcionan una interfaz de sistema de archivos a través de la cual los clientes pueden crear, modificar, leer y borrar archivos.

- Soportan unidades de datos de menor tamaño que los archivos, como páginas, tuplas u objetos.
- Proporcionan facilidades de indexación de los datos, así como facilidades de transacción, de modo que los datos nunca se quedan en un estado inconsistente si falla una máquina cliente o un proceso.

Así como podemos encontrar diferentes tipos de servidores, también se cuenta con dos tipos de arquitecturas cliente/servidor, estas son:

Arquitectura de 2 Capas.

Consta de tres componentes distribuidos en dos capas. Los tres componentes son:

- Interfaz de usuario.
- Administración del procesamiento.
- Administración de la Base de Datos.

Primera Capa. Cliente	Tareas Interfaz de Usuario Lógica y reglas de Procesamiento
Segunda Capa Servidor de Base de Datos	Tareas Validación del Servidor Acceso a la Base de Datos

Arquitectura de 2 Capas

Limitaciones:

- El número máximo de usuarios es de 100. Más allá de este número se excede la capacidad de procesamiento.
- No hay independencia entre la interfaz de usuario y los tratamientos, lo que hace delicada la evolución de las aplicaciones.

Arquitectura de 3 Capas

Surge para superar las limitaciones de la arquitectura de 2 capas. La tercera capa (servidor intermedio) está entre la interfaz de usuario (cliente) y el administrador de datos (servidor). La capa intermedia proporciona administración del procesamiento y en ella se ejecutan las reglas y lógica de procesamiento. Permite cientos de usuarios. Esta arquitectura es usada cuando se necesita un diseño cliente/servidor que proporcione incrementar el rendimiento, flexibilidad, mantenibilidad, reusabilidad y escalabilidad mientras se esconde la complejidad del procesamiento distribuido al usuario.

Primera Capa Cliente	Tareas Interfaz de usuario
Segunda Capa Servidor de aplicaciones	Tareas Reglas y lógica de procesamiento de datos
Tercera Capa Servidor de Bases de Datos	Tareas Validación de datos Acceso a la Base de Datos
Arquitectura de 3 Capas	

Limitaciones:

- Construir una arquitectura de 3 capas es complicado.
- Las herramientas de programación que soportan el diseño de arquitectura de 3 capas no proporcionan todos los servicios deseados que se necesitan para soportar un ambiente de computación distribuida.
- La separación de la interfaz gráfica de usuario, la lógica de administración de procesamiento y la lógica de datos no es siempre muy clara²⁴.

VENTAJAS E INCONVENIENTES DE LA ARQUITECTURA CLIENTE/SERVIDOR

VENTAJAS

Las principales ventajas de la arquitectura cliente/servidor son las siguientes:

- Interoperabilidad. Los componentes clave (cliente, servidor y red) trabajan juntos.
- Flexibilidad. La nueva tecnología puede incorporarse al sistema.
- Escalabilidad. Cualquiera de los elementos del sistema puede reemplazarse cuando es necesario, sin impactar sobre otros elementos. Si la base de datos crece, las computadoras cliente no tienen que equiparse con memoria o discos adicionales. Estos cambios afectan solo a la computadora en la que se ejecuta la base de datos.
- Usabilidad. Mayor facilidad de uso para el usuario.
- Integridad de los datos. Entidades, dominios, e integridad referencial son mantenidas en el servidor de la base de datos.
- Accesibilidad. Los datos pueden ser accedidos desde múltiples clientes.
- Rendimiento. Se puede optimizar el rendimiento por hardware y procesos.
- Seguridad. La seguridad de los datos está centralizada en el servidor.

²⁴ González Martín Oscar, Ruiz González Francisco. Arquitecturas de Sistemas de Bases de Datos. Universidad de Castilla la Mancha. 1999/2000.

INCONVENIENTES

- Hay una alta complejidad tecnológica al tener que integrar una alta variedad de productos. El mantenimiento de los sistemas es más difícil pues implica la interacción de diferentes partes de hardware y de software, distribuidas por distintos proveedores, lo cual dificulta el diagnóstico de fallas.
- Requiere un fuerte rediseño de todos los elementos involucrados en los sistemas de información (modelos de datos, procesos, interfaz, comunicaciones, almacenamiento de datos, etc.). Además, en la actualidad existen pocas herramientas que ayuden a determinar la mejor forma de dividir las aplicaciones entre la parte cliente y la parte servidor.
- Es más difícil asegurar un alto grado de seguridad en una red de clientes y servidores que en un sistema con una única máquina centralizada (cuanto más distribuida es la red, mayor es su vulnerabilidad).
- A veces, los problemas de congestión de la red pueden reducir el rendimiento del sistema por debajo de lo que se obtendría con una única máquina. También la interfaz gráfica de usuario puede a veces ralentizar el funcionamiento de la aplicación.
- Existen multitud de costos ocultos (formación en nuevas tecnologías, cambios organizativos, etc.) que encarecen su implantación.
- La arquitectura cliente/servidor es una arquitectura que está en evolución, y como tal no existe estandarización²⁵.

ANÁLISIS DE ALGUNOS SISTEMAS GESTORES DE BASES DE DATOS (DBMS) COMERCIALES.

La finalidad de este punto es mostrar las características de los siguientes DBMS. Comercialmente se mencionan para su funcionamiento en un ambiente de arquitectura cliente/servidor.

POSTGRESQL

Los Sistemas Gestores de Bases de Datos relacionales tradicionales (SGDB) soportan un modelo de datos que consisten en una colección de relaciones con nombre, que contienen atributos de un tipo específico. En los sistemas comerciales actuales, los tipos posibles incluyen numéricos de punto flotante, enteros, cadenas de caracteres, cantidades monetarias y fechas. Está generalmente reconocido que este modelo será inadecuado para las aplicaciones futuras de procesado de datos. El modelo relacional sustituyó modelos previos en parte por su simplicidad. Sin embargo, como se ha mencionado, esta simplicidad también hace muy difícil la implementación de ciertas aplicaciones. Postgresql tiene algunas características que son propias del mundo de las bases de datos orientadas a objetos. De hecho, algunas Bases de Datos comerciales han incorporado recientemente características en las que Postgresql fue pionera.

²⁵ <http://www.jegsworks.com/Lessons-sp/lesson7/lesson7-6.htm>

VENTAJAS

- Instalación llimitada. Con Postgresql, nadie puede demandarlo por violar acuerdos de licencia, puesto que no hay costo asociado a la licencia del software. Esto tiene varias ventajas adicionales:
 - a. Modelos de negocios más rentables con instalaciones a gran escala.
 - b. No existe la posibilidad de ser auditado para verificar cumplimiento de licencia en ningún momento.
 - c. Flexibilidad para hacer investigación y desarrollo sin necesidad de incurrir en costos adicionales de licenciamiento.
- Estabilidad y confiabilidad legendarias. En contraste con muchos sistemas de bases de datos comerciales, es extremadamente común que compañías reporten que Postgresql nunca ha presentado caídas en muchos años de operación de alta actividad.
- Extensible. El código fuente está disponible para todos sin costo. Si el equipo necesita extender o personalizar Postgresql de alguna manera, pueden hacerlo con un mínimo esfuerzo, sin costos adicionales.
- Multiplataforma. Postgresql está disponible en casi cualquier Unix y una versión nativa de Windows está actualmente en estado beta de pruebas.
- Diseñado para ambientes de alto volumen. Postgresql usa una estrategia de almacenamiento de filas llamada MVCC para conseguir una mucho mejor respuesta en ambientes de grandes volúmenes. Los principales proveedores de sistemas de bases de datos comerciales usan también esta tecnología, por las mismas razones.
- Herramientas gráficas de diseño y administración de bases de datos. Existen varias herramientas gráficas de alta calidad para administrar las bases de datos (pgAdmin, pgAccess) y para hacer diseño de bases de datos (Tora, Data Architect)²⁶.

DESVENTAJAS

- Consume bastantes recursos y hace lento el sistema.
- Límite del tamaño de cada fila de las tablas a 8K (se puede ampliar a 32K recompilando, pero con un costo añadido en el rendimiento)
- Es de 2 a 3 veces más lento que otros manejadores de Bases de Datos²⁷.

ORACLE

Oracle Corporation ha sido líder en la introducción de tecnologías avanzadas de bases de datos cliente/servidor; ha dirigido el desarrollo de sus productos específicamente a soportar el diseño, la implementación y la gestión de los sistemas de bases de datos cliente/servidor.

²⁶ <http://advocacy.postgresql.org/advantages/?lang=es>

²⁷ www.postgresql.com

Oracle soporta la arquitectura cliente/servidor; la premisa de la informática cliente/servidor es distribuir la ejecución de una tarea entre varios procesadores de una red. Cada procesador está dedicado a un conjunto de tareas secundarias enfocadas y específicas que realiza mejor. El resultado es una eficiencia incrementada y una efectividad del sistema como conjunto. La división de la ejecución de las tareas entre los procesadores es realizada a través de un protocolo de solicitudes de servicios. Un procesador, el cliente, solicita un servicio de otro procesador, el servidor. La implementación más extendida del procesamiento cliente/servidor implica la separación de la parte de la interfaz del usuario de una aplicación de la parte del acceso a los datos.

En el cliente, o sección de entrada (front-end), de la configuración típica cliente/servidor se encuentra una estación de trabajo operando con una plataforma GUI (Graphical User Interface, Interfaz Gráfica de Usuario), usualmente Windows, Macintosh o Motif. En el back-end (lado del servidor) de la configuración se encuentra un servidor de base de datos, que ha menudo está gestionado por un sistema operativo UNIX, NetWare, Windows NT o VMS.

VENTAJAS.

- Mecanismos de seguridad. Los sofisticados mecanismos de seguridad de Oracle controlan el acceso a los datos sensibles utilizando un conjunto de privilegios. En función del nombre con el que se conectan a la base de datos, a los usuarios se les conceden derechos para consultar, modificar y crear datos.
- Realización de copias de seguridad y recuperación. Oracle proporciona sofisticados procedimientos de realización de copias de seguridad y recuperación de los datos. Las copias de seguridad permiten crear una copia secundaria de los datos de Oracle; los procedimientos de recuperación restauran los datos a partir de una copia de seguridad.
- Control del espacio. Se puede asignar un cierto espacio de disco para el almacenamiento de los datos, y controlar las subsiguientes asignaciones instruyendo a Oracle sobre cuánto espacio reservar para los requerimientos futuros. También tiene una serie de características que fueron diseñadas teniendo en cuenta las necesidades de las bases de datos de muy gran tamaño.
- Conectividad de carácter abierto. Oracle proporciona conectividad hacia y desde paquetes software de otros fabricantes. Utilizando las extensiones a la base de datos Oracle, se puede trabajar con información almacenada con otros sistemas de bases de datos, como Sybase o Access. También se pueden almacenar los datos en la base de datos de Oracle y acceder a ellos desde otros paquetes software, como Visual Basic, Powerbuilder, o SQL Windows.
- Herramientas de desarrollo. El servidor Oracle, al que normalmente se denomina motor de la base de datos, funciona con un amplio conjunto de herramientas de desarrollo, herramientas de consulta para usuario final, aplicaciones comerciales y herramientas de gestión de la información de ámbito corporativo.

- Mecanismos de integridad. Si se produce cualquier tipo de fallo mientras un usuario está cambiando los datos en una base de datos, ésta tiene la capacidad de deshacer o cancelar cualquier transacción sospechosa²⁸.

DESVENTAJAS

- El mayor inconveniente de Oracle es quizás su precio. Incluso las licencias de Personal Oracle son excesivamente caras. Otro problema es la necesidad de ajustes. Un error frecuente consiste en pensar que basta instalar el Oracle en un servidor y realizar la instalación directamente de las aplicaciones clientes. Un Oracle mal configurado puede ser desesperantemente lento.
- También es elevado el costo de la formación, y sólo últimamente han comenzado a aparecer buenos libros sobre asuntos técnicos distintos de la simple instalación y administración.
- No se cuenta con la suficiente documentación para su manejo.

ACCESS

Microsoft Access es un sistema interactivo de administración de bases de datos para Windows. Access tiene la capacidad de organizar, buscar y presentar la información resultante del manejo de sus bases de datos.

Access es gráfico, por lo que aprovecha al máximo la potencia gráfica de Windows, ofreciendo métodos usuales de acceso a los datos y proporcionando métodos simples y directos de trabajar con la información. Facilita la administración de datos, ya que sus posibilidades de consulta y conexión le ayudan a encontrar rápidamente la información deseada, cualquiera que sea su formato o lugar de almacenamiento.

Con Access es posible producir formularios e informes sofisticados y efectivos, así como gráficos y combinaciones de informes en un solo documento. Access permite lograr un considerable aumento en la productividad mediante el uso de los asistentes y las macros. Estos permiten automatizar fácilmente muchas tareas sin necesidad de programar²⁹.

VENTAJAS

- Tiene un sistema de almacenamiento de datos.
- Permite enlazar información relacionada fácilmente.
- Puede trabajar con datos procedentes de otras fuentes, incluyendo muchos programas populares de bases de datos para PC y muchas bases de datos SQL sobre servidores, minicomputadoras y computadoras centrales.
- Cuenta con un sistema de desarrollo de aplicaciones sofisticado para el sistema operativo de Microsoft, que facilita el uso extensivo de la información sobre los

²⁸ González Martín Oscar, Ruiz González Francisco. Arquitecturas de Sistemas de Bases de Datos. Universidad de Castilla la Mancha. 1999/2000.

²⁹ http://www.microsoft.com/spain/office/access/access_5.asp

datos, cualquiera que sea el origen de los mismos, ayudándonos a construir aplicaciones fácilmente. Esto es, no es necesario escribir ningún código complejo en el sentido de la programación clásica.

DESVENTAJAS

- Si se quiere actualizar la versión con la que se esté trabajando, se corre el riesgo de afectar el funcionamiento de la aplicación, con lo cual podría perderse la información o no respetar la estructura de la base de datos con la que se trabaja.
- No tiene el rendimiento adecuado al manejar grandes cantidades de información, lo cuál se convierte en un serio problema de seguridad.
- Conforme aumenta el número de usuarios la red se vuelve lenta.
- Es sencillo quebrantar la seguridad de la información almacenada en una base de datos creada en Access³⁰.

POWER BUILDER

En PowerBuilder se vislumbran tres estrategias de desarrollo claramente diferenciadas, sin menoscabo de otros aspectos interesantes del mismo.

Estas estrategias de desarrollo son las siguientes:

- Cliente/Servidor 2 capas.
- Cliente/Servidor distribuido.
- Internet.

Sobre el desarrollo de aplicaciones Cliente/Servidor en dos capas PowerBuilder proporciona, a través de la interfaz ODBC y de la utilización de los drivers nativos de los motores más importantes existentes en el mercado, los medios necesarios para el desarrollo de aplicaciones Cliente/Servidor de dos capas.

Soporta el uso de procedimientos almacenados, una de las herramientas más potentes existentes en los motores de bases de datos más avanzados.

Como característica adicional, posee un tipo de objeto, que proporciona un acceso inmediato, transparente y encapsulado, permitiendo a la vez, la presentación y/o manipulación de la información seleccionada, de una manera eminentemente sencilla, y con un costo mínimo de desarrollo.

ANÁLISIS DE LA RED ACTUAL EN USECAD

La conexión que existe entre la Dirección General de Servicios de Cómputo Académico (DGSCA) y la Facultad de Ingeniería está hecha con base en una **Topología de tipo Bus**, la cual tiene las características mencionadas en la parte superior de este capítulo.

³⁰ <http://www.jegsworks.com/Lessons-sp/lesson1-2/lesson2-4database.htm>

De forma interna, la Facultad de Ingeniería trabaja con una **Topología de tipo Estrella** pues esta configuración se utiliza para actividades que manejan grandes cantidades de información que sufre rápidos cambios, el cual es nuestro caso; además de que si falla un nodo que no sea el principal, no se ve afectada la funcionalidad de la red, es una de las topologías más rápidas en situaciones de tráfico pesado. Características de suma importancia para realizar de manera segura el envío de información a la USECAD.

Cabe mencionar que el manejador de Bases de Datos, que utiliza USECAD, se encargará de la **Seguridad y encriptación** de la información durante el envío de la misma. Al enviar la información el DBMS encripta la información haciendo uso de sus códigos de encriptación y cuando la información llegue a la máquina destino, quitará el encriptamiento usando nuevamente sus códigos para ello.

Para que esta red pueda funcionar de forma adecuada se utiliza el protocolo de comunicaciones TCP/IP; cabe mencionar que dentro de la Facultad de Ingeniería se utilizan, además del protocolo anterior, otros protocolos de comunicación que permiten realizar la comunicación deseada, entre ellos se encuentra el protocolo NetBios.

CAPÍTULO 6 PRESENTACIÓN DE LA PROPUESTA

A continuación se presentan los **modelos y diagramas** para la descripción del funcionamiento de la propuesta. Estos modelos están basados en los que utiliza la USECAD para el manejo de su información y, para el desarrollo de nuestra propuesta adecuaremos dichos modelos y diagramas.

Se presenta el modelo conceptual de la base de datos que proponemos para realizar las inscripciones de los laboratorios a través del diagrama Entidad – Relación; así como el diagrama de flujo de información de la propuesta que estamos desarrollando y los requerimientos del hardware para realizarla.

Por último describimos la construcción de una interfase en Power Builder como propuesta para el acceso a la base de datos de la USECAD y a la administración interna de la base de datos que utilizará el Departamento de Ingeniería Mecánica.

DIAGRAMA DE FLUJO PROPUESTO DEL PROCESO 2.1.6

El origen de este diagrama se presenta en el Capítulo 1 "Planteamiento del problema".

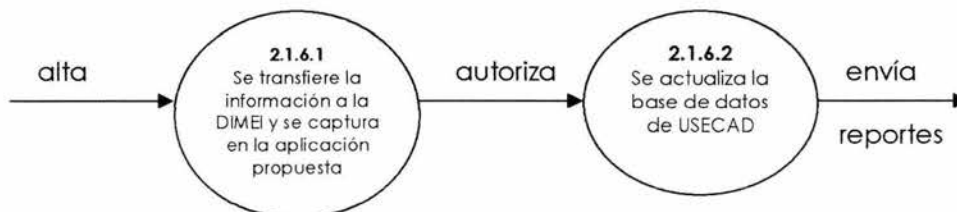
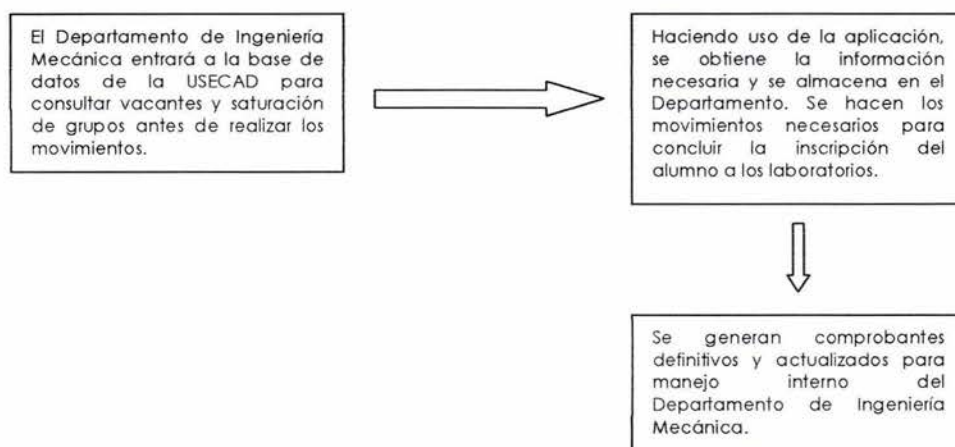


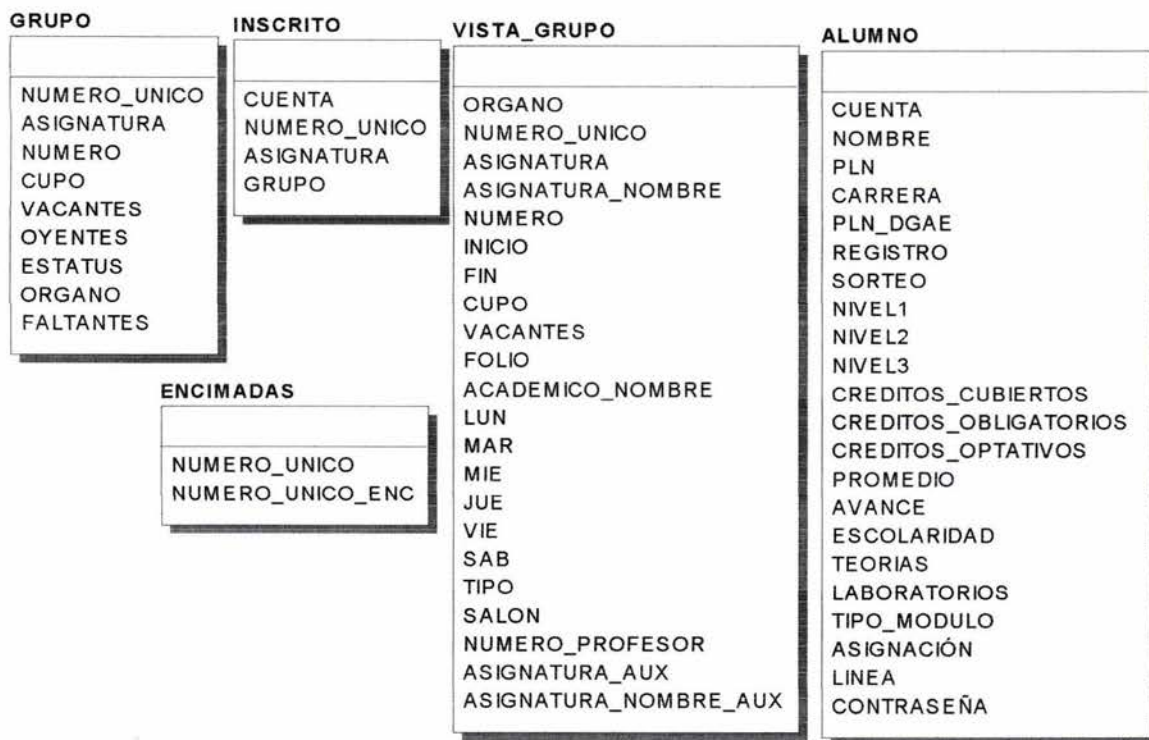
DIAGRAMA DE FLUJO PROPUESTO DE LA INFORMACIÓN DE MOVIMIENTOS DE INSCRIPCIÓN DE LABORATORIOS EN EL DEPARTAMENTO DE INGENIERÍA MECÁNICA (PROCESO INTERNO)



A continuación, mostramos las tablas que se deben consultar, si es que se realiza el desarrollo de nuestra propuesta, estas tablas pertenecen a la base de datos de USECAD, las cuales contienen la información requerida por el Departamento de Ingeniería Mecánica para realizar los movimientos solicitados por los alumnos.

Tablas de USECAD a consultar.

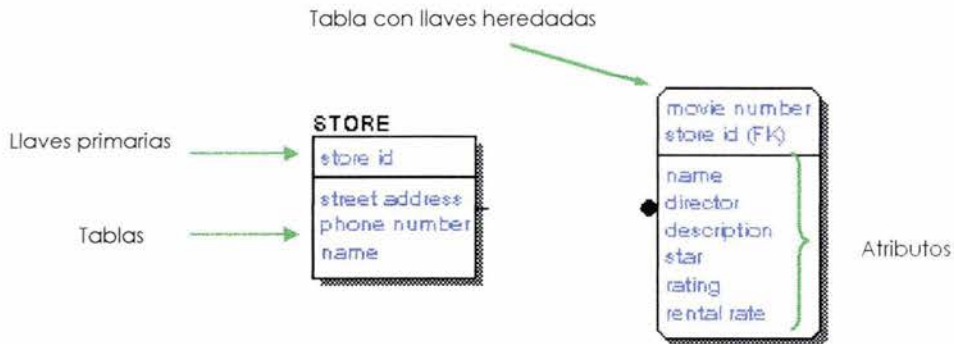
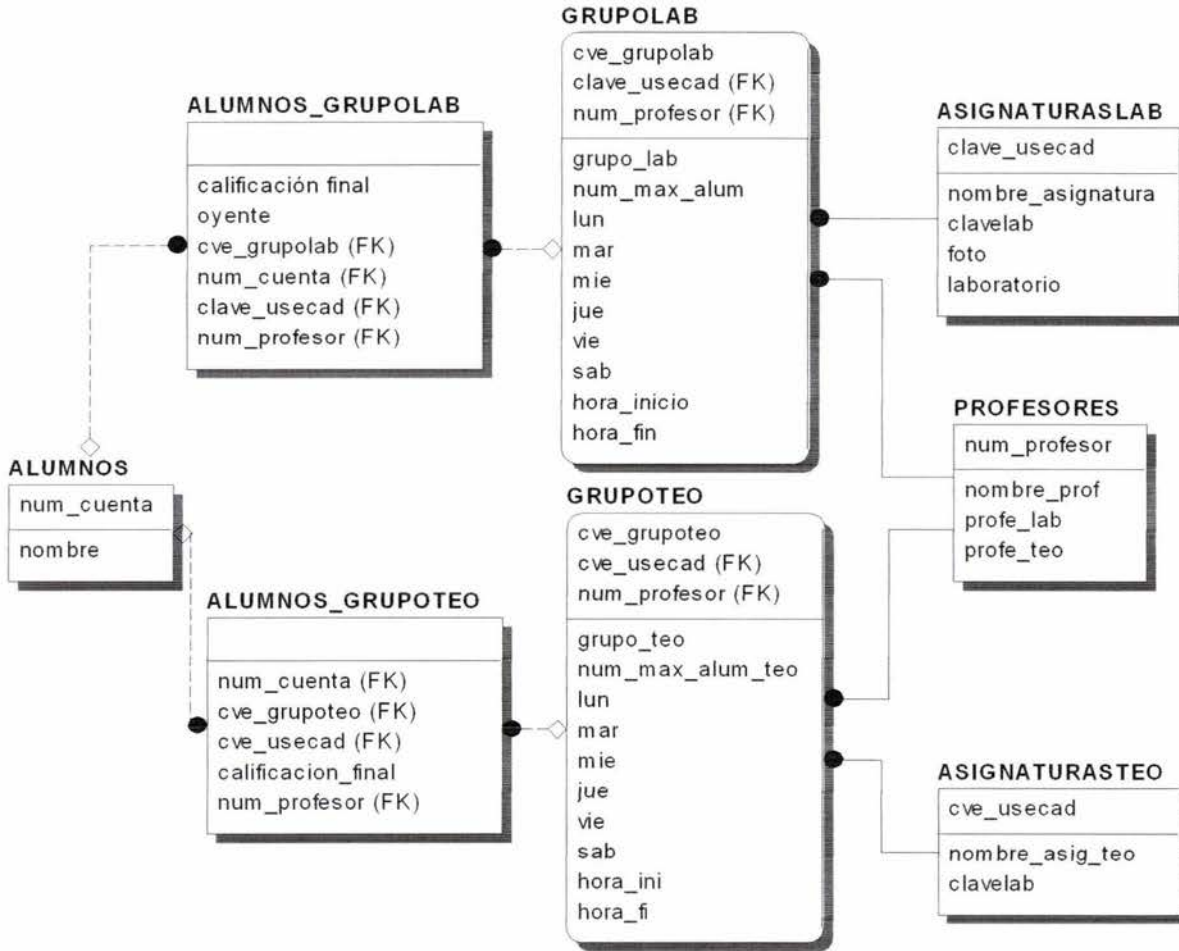
La base de datos que se va a consultar en el departamento de la USECAD se llama **cleo-fi**. Las tablas que son de utilidad para nuestro caso son las siguientes:



No todos los campos de estas tablas son de utilidad para el funcionamiento de la aplicación; solamente seleccionaremos los campos necesarios que compaginen con la información que se muestra en nuestro diagrama Entidad – Relación, el cual muestra el modelo conceptual del funcionamiento de la base de datos de la propuesta que estamos realizando.

Es importante mencionar que por no contar con los permisos requeridos para acceder a USECAD, simularemos la conexión haciendo uso de una base de datos interna.

MODELO ENTIDAD RELACIÓN DE LA PROPUESTA



- Relación uno a muchos (debe) ———●
- Relación uno a muchos (puede) - - - -●

REQUERIMIENTOS DE HARDWARE

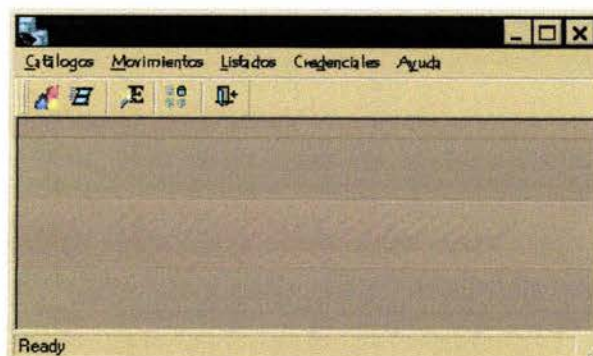
Contar con una computadora que contenga por lo menos las siguientes especificaciones:

- Sistema operativo Microsoft Windows versión 98.
- Memoria RAM 64 MB.
- Espacio en disco duro 500 MB.
- Ratón.
- Procesador Pentium III.
- Tarjeta gráfica.
- Tarjeta de red.
- Unidad de CD.
- Unidad de 3 ½ pulgadas.
- Power Builder versión 8.
- Access.

INTERFASE PROPUESTA EN POWER BUILDER

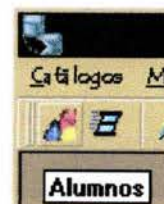
Esta es la ventana de inicio al ejecutar la aplicación que contiene el siguiente menú:

- Catálogos.
- Movimientos.
- Listados.
- Credenciales.
- Ayuda.



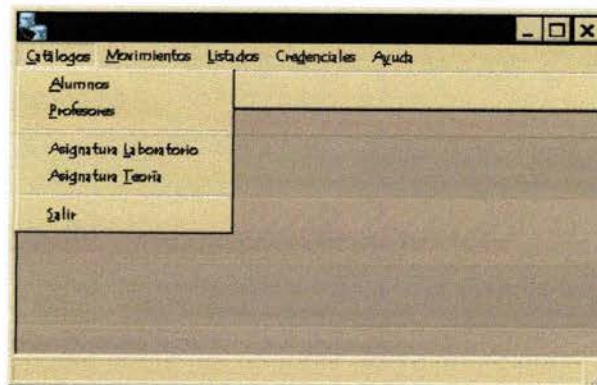
También se cuenta con íconos de acceso directo a las opciones más utilizadas, estas son:

- profesores.
- alumnos
- asignatura laboratorio.
- asignatura teoría.
- salir del sistema.



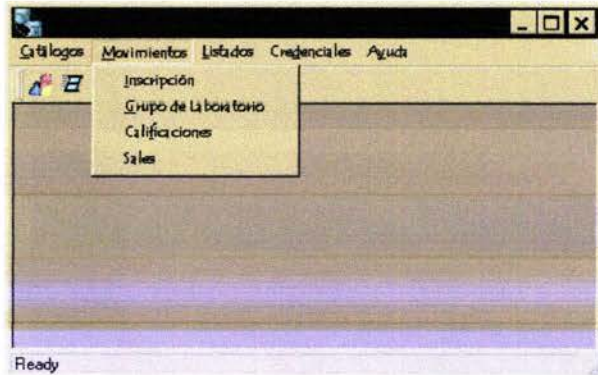
Este menú nos sirve para consultar información de:

- alumno
- profesor.
- asignatura de laboratorio.
- asignatura de teoría.
- salir del sistema.



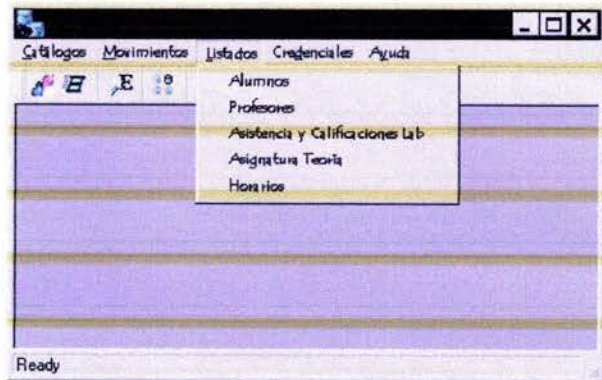
Este menú nos permite llevar a cabo los movimientos requeridos por el alumno como son:

- inscripciones a los grupos de laboratorio.
- al Departamento de I. M. para almacenar las calificaciones obtenidas en el semestre.



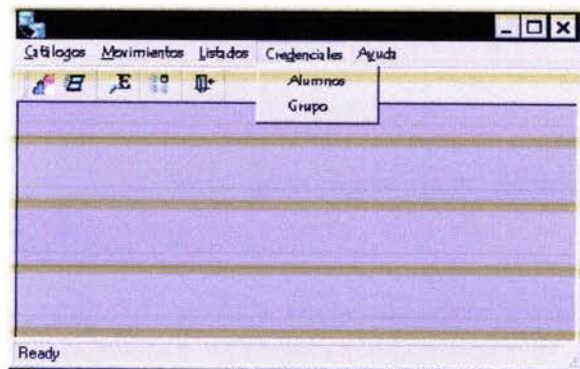
A través de este menú obtenemos listados de:

- grupos a los que está inscrito el alumno.
- grupos a los que un profesor imparte clase y sus listados para toma de asistencia y registro de calificaciones finales.
- proporcionar los listados de calificaciones a los profesores de teoría.
- horarios que se programan en cada semestre.



Este menú permite realizar el reporte para la impresión de credenciales. Éste puede llevarse a cabo de las siguientes formas:

- por alumno.
- en bloques por grupo.



CAPÍTULO 7

DESCRIPCIÓN DE PRUEBAS Y POSIBLES RESULTADOS

En el presente capítulo describiremos algunos aspectos de la instrumentación del desarrollo de la programación, así como lo que es la programación modular y analizaremos las diferentes pruebas que deben realizarse a un software en general, para verificar que éste realiza las actividades para las que fue programado, aunado a ello, se verifica que el funcionamiento es el requerido por el usuario y que posee seguridad para el manejo de información.

ASPECTOS DE LA INSTRUMENTACIÓN DE LA PROGRAMACIÓN Y PROGRAMACIÓN MODULAR

La fase de instrumentación del desarrollo de la programación tiene que ver con la traducción de las especificaciones del diseño a código fuente. La sencillez, claridad y elegancia son los sellos de los buenos programas³¹.

La claridad del código fuente se mejora mediante técnicas de codificación estructurada, buen estilo de codificación, documentos adecuados de apoyo, buenos comentarios internos, y por las características que proporcionan los lenguajes de programación modernos.

El objetivo de la codificación estructurada es linealizar el flujo de control a través de un programa de computadora, de modo que la ejecución siga a la secuencia en que está escrito el código. La estructura se parece a un texto escrito que mejora la legibilidad del código, facilita la verificación formal de programas y el flujo de control lineal se puede lograr restringiendo el conjunto de construcciones de programa permitidas a formatos de entrada única y salida única.

Programación modular.

La programación modular se basa en la descomposición en módulos de un programa o un sistema. La unidad de la modularidad de los procedimientos en los que se resuelve un problema mediante la programación, dará como resultado que el programa diseñado para un propósito determinado, funcione de acuerdo a su diseño y especificación del cliente.

A continuación se hace un análisis de los diferentes tipos de pruebas con el objetivo de determinar las pruebas de software para la aplicación propuesta de este trabajo.

PRUEBAS

Prueba de Software

Es el proceso de revisar que un sistema sea correcto. Este proceso se describe mediante el término de **validación** y **verificación**: dos actividades separadas que aseguran que el sistema que se entrega cumple con los requisitos del usuario.

³¹ Fairley Richard. Ingeniería de Software. Addison Wesley. 1988.

La **validación** es el proceso de asegurar que un sistema cumpla con los requisitos de un usuario y la **verificación** es asegurar que una actividad, por ejemplo, el diseño del sistema, se lleve a cabo correctamente³².

EL PROCESO DE PRUEBA

Los sistemas grandes se componen de subsistemas formados por módulos que, a su vez, pueden componerse de procedimientos.

El proceso de prueba, al igual que el de programación, debe avanzar en etapas, siendo cada una de ellas la continuación lógica de la etapa anterior.

En el proceso de prueba se pueden identificar cinco etapas:

1. **Prueba de funciones.** La prueba de funciones o de unidades es el nivel básico en donde se prueban las funciones que componen un módulo para garantizar que operan de manera correcta. Las funciones no deben depender de otras funciones de su mismo nivel.
2. **Prueba de módulos.** Un módulo se compone de varias funciones que pueden cooperar entre sí. Después de haber probado cada función individual, es necesario probar un módulo como una entidad aislada, sin la presencia de otros módulos del sistema.
3. **Prueba de subsistemas.** Esta prueba es el siguiente paso del proceso en el cual los módulos se agrupan para formar subsistemas. Puesto que los módulos cooperan y se comunican, la prueba de subsistemas se debe centrar en la prueba de las interfaces de aquellos, dando por supuesto que los módulos son correctos.
4. **Prueba del sistema.** La prueba del sistema se lleva a cabo cuando se integran los subsistemas para conformar el sistema completo.
5. **Prueba de aceptación.** Esta prueba se efectúa con datos reales: la información con la que el sistema deberá operar.

MODULARIDAD EN PRUEBAS

Un enfoque para efectuar la comprobación del sistema, es introducir los módulos ya probados de manera gradual, uno a la vez.

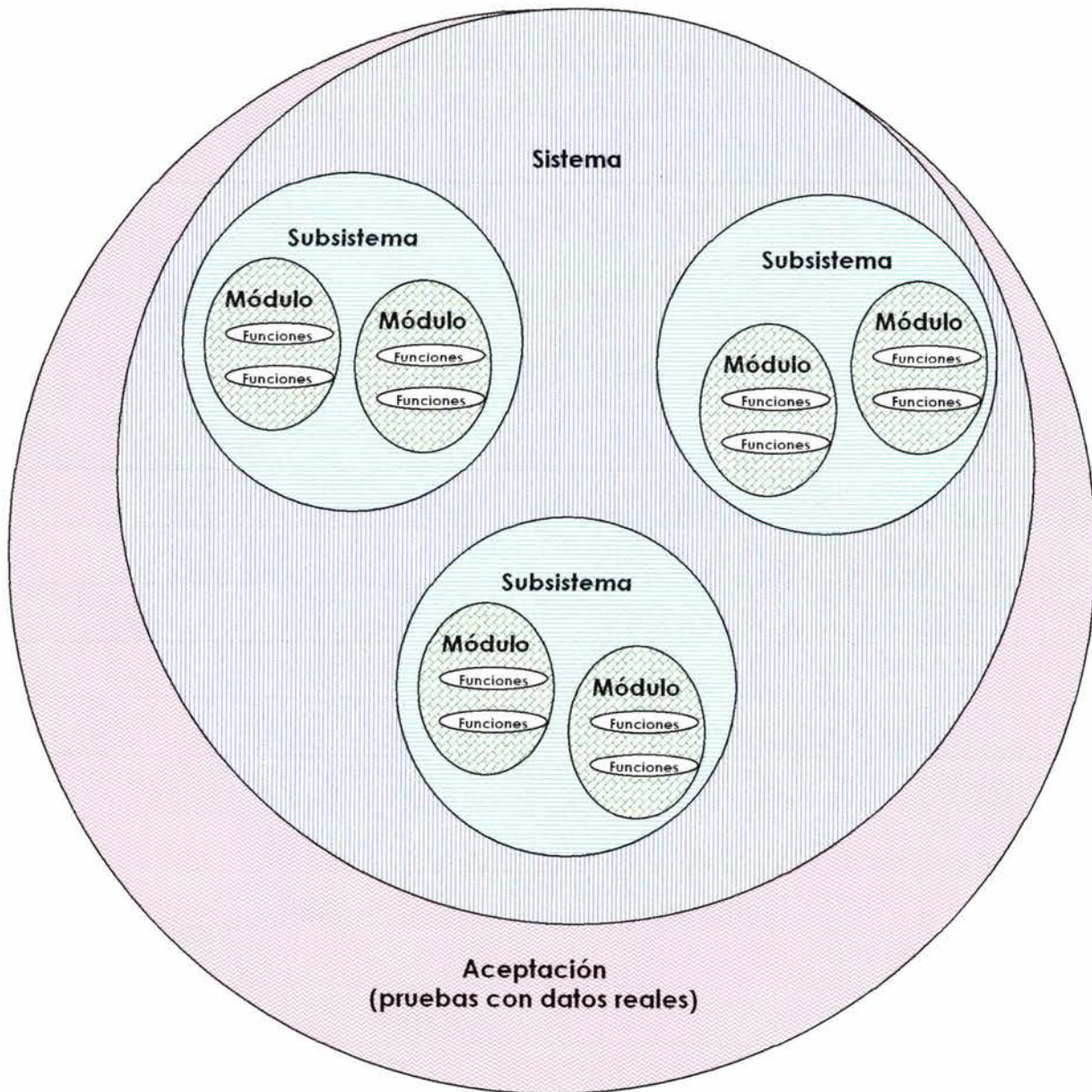
El sistema debe empezar como un módulo, y así debe probarse por medio de los casos de prueba adecuados. Una vez que la prueba de este módulo es satisfactoria, se puede introducir el segundo módulo y hacer las pruebas. El proceso continúa hasta que todos los módulos se han integrado en un sistema completo. Si en alguna etapa de este proceso se introduce un módulo en el que las pruebas no detectaron errores en el sistema, y se hallan errores, es

³² Molino Enzo, Mora José Luis. Introducción a la Informática. Trillas. 1982. 1ª Edición.

probable que éstos se deban a la inclusión del nuevo módulo, de ese modo se localiza la fuente del error y se simplifica la tarea de encontrarlo.

En la siguiente figura podemos representar mediante conjuntos como se agrupan las pruebas por etapas.

Proceso de prueba



-Prueba de funciones	
-Prueba de módulos	
-Prueba de subsistemas	
-Prueba del sistema	
-Prueba de aceptación	

PRUEBAS DE UNIDAD

Las pruebas de unidad comprenden el conjunto de pruebas efectuadas por un programador individual, antes de la integración de la unidad en un sistema más grande. La situación se ilustra como sigue:

Codificación y depuración → pruebas de unidad → pruebas de integración

Esta prueba debe ser mucho más minuciosa que el examen al que se someterá cuando la unidad se integre en el producto de software en desarrollo. Hay cuatro categorías de pruebas que, por lo común, efectuará un programador a una unidad de programa:

- Pruebas funcionales
- Pruebas de desempeño
- Pruebas de tensión
- Pruebas de estructura

Los casos de *prueba funcional* implican ejercitar el código con valores nominales de entrada para los cuales se conocen los resultados esperados, además de valores límites.

Las *pruebas de desempeño* determinan la cantidad de tiempo de ejecución empleado en varias partes de la unidad, la eficiencia global del programa, el tiempo de respuesta, y la utilización de dispositivos por la unidad de programa.

Las *pruebas de tensión* son aquellas diseñadas para romper, en forma intencional, la unidad de programa. Éstas se relacionan con ejercitar la lógica interna de un programa y recorrer rutas de ejecución particulares. Algunos autores se refieren colectivamente a las pruebas funcionales, de desempeño y de tensión como pruebas de “**caja negra**”, mientras que a las de estructura las denominan pruebas de “**caja blanca**” o “**crystal**”.

Las *pruebas estructurales* deciden cuales rutas ejercitar, obtienen los datos de prueba para ejercitar esas rutas, determinan el criterio de cobertura de la prueba que se usará, ejecutan los casos de prueba y miden la cobertura de la prueba lograda cuando se ejercitan esos casos.

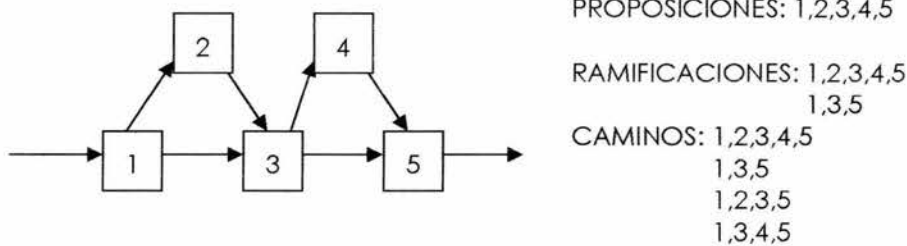
Aún cuando fuera posible probar con éxito todas las rutas a través de un programa, la corrección no estaría garantizada. Un error de ruta ignorada ocurre cuando por accidente se omiten una proposición de ramificación y el cálculo asociado; tales errores sólo se pueden detectar por casos de prueba funcionales derivados de las especificaciones de requisitos.

Los errores se clasifican como **errores por ignorar una ruta**, **computacionales**, y **de dominio**. Con cálculos lineales se desea expresar que todos los cálculos son funciones lineales de las variables de entrada, cuando los valores simbólicos de entrada se propongan a través del programa.

La teoría de pruebas de dominios necesita, que cada límite de cada dominio se determine por un predicado lineal que tenga sólo un operador relacional.

La teoría de pruebas, basada en modelos simplificadores, proporciona una comprensión de las dificultades que se presenten en las pruebas de programas reales.

En la práctica, hay *tres medidas de cubrimiento* que suelen usarse en la prueba de unidad: **cobertura de proposiciones, de ramificaciones y de rutas lógicas**. Al emplear la primera como criterio de terminación de la prueba, el programador está intentando hallar un conjunto de casos de prueba que ejecutarán cada proposición del programa por lo menos una vez. En la segunda el programador intenta encontrar un conjunto de casos de prueba que ejecute cada proposición de las ramificaciones en cada dirección por lo menos una vez. La cobertura de rutas lógicas reconoce que el orden en que se ejecutan las ramas durante una prueba es un factor importante para determinar el resultado de la prueba. Estas medidas se ilustran en la siguiente figura.



Por lo común, las pruebas funcional, de desempeño y de tensión basadas en los requisitos funcionales y en la intuición de un programador lograrán de un 60 a 70% de la cobertura de las proposiciones. Por lo tanto, agregar pruebas de ramificación es una mejora significativa en la cobertura de las pruebas.

Una técnica empleada a menudo durante las pruebas de unidad es la medición de la cobertura de ruta lograda por las pruebas funcional, de desempeño y de tensión; además agregar casos de prueba adicionales hasta que se logra el nivel deseado de cobertura de ramificación.

PRUEBAS DEL SISTEMA

Este tipo de pruebas implica dos clases de actividades: *pruebas de integración* y *de aceptación*.

Las pruebas de aceptación implican la planeación y ejecución de pruebas funcionales, de desempeño y de tensión para verificar que el sistema realizado satisfaga sus requisitos. Las pruebas de integración se realizan posteriormente a las pruebas de unidad y su foco de atención es el diseño y la construcción de la arquitectura del software.

a) Pruebas de integración.

La integración ascendente es empleada para integrar los componentes de un sistema de software; consiste en pruebas de unidad, seguidas por pruebas de subsistemas y luego por pruebas del sistema completo. Las primeras tienen el objetivo de descubrir errores en los módulos individuales del sistema.

El propósito principal de las pruebas de subsistemas es verificar la operación de las interfaces entre módulos en el subsistema. Se deben probar tanto las interfaces de control como las de datos.

Las pruebas del sistema se relacionan en las interfaces, la lógica de decisión, el flujo del control, los procedimientos de recuperación, la eficiencia global, la capacidad, y las características de tiempo del sistema entero.

b) Pruebas de aceptación

Las pruebas de aceptación implican la planeación y la ejecución de pruebas funcionales, de desempeño y de tensión para demostrar que el sistema implantado satisface sus requisitos. Con frecuencia, se corren dos conjuntos de pruebas de aceptación: aquellas desarrolladas por el grupo de control de calidad y las que hace el cliente. Por ejemplo, un compilador puede probarse para determinar el efecto del sobre flujo en la tabla de símbolos, o un sistema de tiempo real se puede probar para conocer el efecto del arribo simultáneo de muchas interrupciones de alta prioridad.

Las pruebas de aceptación incorporan casos de prueba desarrollados durante las pruebas de unidad y de aceptación. Se añaden casos de prueba adicionales para lograr el nivel deseado de las pruebas funcional, de desempeño y de tensión del sistema completo. Entre las herramientas de especial importancia durante las pruebas de aceptación se hallan un analizador de cobertura de prueba y otro de tiempos, además un verificador de los estándares de codificación.

Un analizador de la cobertura de la prueba registra las rutas de control seguidas por cada caso de prueba. Sin esta herramienta sería imposible establecer la extensión lograda de la cobertura de la prueba.

Un analizador de tiempos informa el tiempo empleado en varias regiones del código fuente bajo diferentes casos de prueba. Es normal que un programa utilice de un 80 a un 90% del tiempo de ejecución en 20% o menos del código. Estas regiones de código son áreas para concentrarse y mejorar el desempeño del sistema.

PRUEBAS DE VALIDACIÓN

Tras la culminación de las pruebas de integración, el software está completamente ensamblado como un paquete; se han encontrado y corregido los errores de interfaz pueden comenzar una serie final de pruebas de software: las pruebas de validación. La validación puede definir varias formas, pero una simple definición es que la validación se consigue cuando el software funciona de acuerdo a las expectativas razonables del cliente.

PRUEBAS ALFA

La prueba alfa es conducida por un cliente en el lugar de desarrollo. Se usa el software de manera natural, con el encargado de desarrollo y registrando errores y problemas de uso. Estas pruebas se llevan a cabo en un entorno controlado.

PRUEBAS BETA

La prueba beta se lleva a cabo en uno o más lugares de clientes por los usuarios finales del software. A diferencia de la prueba alfa, el encargado del desarrollo, normalmente, no está presente. La prueba beta es una aplicación "en vivo" del software en un entorno que no puede ser controlado por el equipo de desarrollo. El cliente registra todos los problemas (reales o imaginarios) que encuentra y los informa a intervalos regulares. Como resultado, el equipo de desarrollo lleva a cabo modificaciones y así prepara una versión del producto para toda la base de clientes.

DESARROLLO DE PRUEBAS A LA APLICACIÓN PARA LA INSCRIPCIÓN A LOS LABORATORIOS DEL DEPARTAMENTO DE INGENIERÍA MECÁNICA

Para realizar las pruebas a nuestro software, seguimos los cinco pasos del proceso de prueba de software de la siguiente manera:

Pruebas de Unidad del Módulo 1

Pruebas de caja negra.

1. **Pruebas funcionales:** Dar de alta, baja y cambio de grupo a varios alumnos en diferentes grupos de laboratorio; comprobar el funcionamiento de la aplicación consultando listas de grupo antes y después de los movimientos.
2. **Pruebas de desempeño:** Verificar la eficiencia de la aplicación a través del tiempo de respuesta que tienen cada solicitud.
3. **Pruebas de tensión.** Comprobar que la aplicación al momento de ejecutar las pruebas estructurales las realice y no cause problemas a la estructura de la base de datos interna, externa y a la aplicación misma.

Pruebas de caja blanca.

1. **Pruebas estructurales:** Probar que la aplicación realice las actividades descritas por el usuario el capítulo del planteamiento del problema, y verificar los siguientes casos: saturación de grupos, cancelación de grupos con alumnos inscritos; ingresar datos erróneos, como números de cuenta con más y menos dígitos de los aceptados, caracteres alfabéticos en lugar de numéricos, alta de alumnos en grupos que no existen, alta de alumnos en laboratorios cuando aún no cuentan con la inscripción al grupo de teoría correspondiente.

Pruebas de Unidad del Módulo 2

Pruebas de caja negra.

1. **Pruebas funcionales:** Obtener los permisos de acceso a la base de datos de la USECAD.
2. **Pruebas de desempeño:** La aceptación, el rechazo ó el error de conexión se puede tomar como medida de prueba al acceso del servidor de USECAD.
3. **Pruebas de tensión.** Realizar movimientos no autorizados, como tratar de alterar la información de la base de datos (alterar registros o campos, eliminarlos o generar otros).

Pruebas de caja blanca.

1. **Pruebas estructurales:** Ver si se obtiene respuesta de comunicación. Si no es así, verificar los medios físicos (funcionamiento de tarjeta de red, servidores locales, cable de red) o los medios lógicos para realizarla (configuración incorrecta de los equipos utilizados para la comunicación).

Pruebas del sistema.

Pruebas de integración

Una vez establecida la comunicación entre el Departamento de Ingeniería Mecánica y USECAD, se verificará que los datos importados sean los solicitados para realizar los movimientos requeridos (altas, bajas, cambios de grupo, cancelaciones, generación de listados y credenciales).

Pruebas de Validación

la validación se consigue si al hacer las pruebas de inscripción la aplicación funciona de acuerdo a los requisitos del departamento de Ingeniería Mecánica.

Pruebas de aceptación.

Este tipo de pruebas se realizan con datos reales, es decir, durante un periodo de inscripciones ordinarias.

ALCANCE DE ESTE TRABAJO.

La entrega de este trabajo corresponde al desarrollo de las pruebas del **subsistema uno**. Como medida de pruebas del **subsistema dos** simulamos una conexión a una base de datos ya que no contamos con los permisos de acceso a la base de datos de USECAD.

La etapa de realización de pruebas del **subsistema uno**, nos permite verificar el funcionamiento de la base de datos en el departamento de Ingeniería Mecánica. Al controlar directamente su

CAPÍTULO 7. DESCRIPCIÓN DE PRUEBAS Y POSIBLES RESULTADOS

funcionamiento estamos entregando una aplicación con prueba de validación **alfa** ya que cubre las necesidades de inscripción interna en el departamento.

Las pruebas de validación **beta** corresponderán al uso de la aplicación en otra área que requiera el control de la inscripción de laboratorios o algún otro usuario que lo quiera utilizar.

RESULTADOS ESPERADOS

Esta propuesta puede automatizar el proceso de inscripciones a los laboratorios del departamento de Ingeniería Mecánica, permitiendo realizar los movimientos necesarios para finalizar la inscripción de los alumnos de forma segura e inmediata, se puede generar la impresión de listados preliminares y definitivos, así como las credenciales de cada uno de los alumnos.

Los resultados esperados pueden ser muy satisfactorios pues se cuenta con los siguientes beneficios:

- Listados de grupo de laboratorio con formato para asistencia de alumnos.
- Listados de grupos de laboratorio con formato para calificaciones.
- Listados de grupo de laboratorio con resultados de calificaciones.
- Listados de grupos de teoría con resultados de calificaciones
- Credenciales para los alumnos.
- Listados de relaciones de profesores y grupos de laboratorio asignado.
- Listados de relaciones de profesores y grupos de teoría asignados.

Una vez teniendo los permisos necesarios y la conexión con USECAD, podemos conocer la situación de los alumnos (respecto a los laboratorios inscritos de forma ordinaria). De igual forma podemos realizar internamente los procesos de inscripciones, altas, bajas de alumnos y grupos.

CONCLUSIONES

Según el diccionario: ingeniería "es la aplicación de las ciencias fisicomatemáticas a la invención perfeccionamiento y utilización de la técnica industrial. Conjunto de los estudios que permiten determinar, para la realización de una obra o un programa de inversiones, las orientaciones más deseables, la mejor concepción, las condiciones de rentabilidad óptimas y los materiales y procedimientos más adecuados"³³.

Los conocimientos que adquirimos a lo largo de la carrera de ingeniería en computación, nos proporcionaron los conocimientos necesarios para poder analizar problemas y encontrarles solución a través de la tecnología de hardware y software.

En el caso específico de esta tesis, analizamos el problema que el Departamento de Ingeniería Mecánica tiene al realizar su proceso de inscripción de alumnos. Nosotros estudiamos la manera de obtener la información que necesita el departamento; y elaboramos una propuesta que puede agilizar el proceso de una manera eficiente.

Al implementar la aplicación, podremos ver que se obtendrá como resultado final lo siguiente:

- Una aplicación que permita llevar a cabo los movimientos que los alumnos requieran para finalizar su inscripción semestral a los laboratorios que pertenecen a la carrera de Ingeniería Mecánica, Industrial y Mecatrónica; así mismo, puede llevarse a cabo la impresión de listados de asistencia, calificaciones y credenciales de los alumnos registrados.
- De igual forma, el Coordinador de los laboratorios del Departamento de Ingeniería Mecánica puede mantener un registro del semestre anterior de los alumnos que hayan acreditado el laboratorio.

Algunos de los beneficios resultarán, la aportación de la aplicación al departamento de Ingeniería Mecánica, ya que engloba en todo el proceso, un mejoramiento continuo, progresivo, y en consecuencia, ayuda a mantener un nivel operacional del programa.

Se logrará agilizar el sistema de inscripciones a los laboratorios del departamento de Ingeniería Mecánica en los puntos siguientes:

1. Procesamiento inmediato de las listas de alumnos.
2. Existe una adecuada distribución de la información.
3. Existe menor tiempo de ajuste en listas.
4. Facilidad para inscribir alumnos y buscar en que grupos están inscritos.
5. Disposición para realizar consultas a grupos y profesores.

Éstos son los principales beneficios de la aplicación que se sugiere para llevar a cabo el proceso de inscripción a los laboratorios del departamento de Ingeniería Mecánica.

Para poder llevar a cabo el desarrollo de este trabajo, se tuvo que echar mano de los conocimientos adquiridos a lo largo de la carrera, y la investigación bibliográfica para ampliarlos; además de tomar cursos y asesorías.

³³ _____." Pequeño Larousse Ilustrado". México. 1982. Pág. 578.

El uso de diagramas de flujo de información, nos permite delimitar los requisitos de una persona o empresa al momento de usar un software. Los diagramas empleados en esta tesis permitieron ver las necesidades que tiene el departamento de Ingeniería Mecánica al realizar una inscripción cada semestre; así como plantear esta solución para que cuente con la información actualizada.

La Ingeniería del Software es desarrollar la solución de un problema por etapas. Cada etapa tiene un ascenso a otra etapa y una vez cubiertas todas, nos da como resultado un programa tras una propuesta que se ajusta a las necesidades del coordinador de laboratorios del Departamento de Ingeniería Mecánica.

En pocas palabras, el uso de los conceptos de Ingeniería, desde matemáticas básicas hasta la elaboración y construcción de un programa nos permitió proponer una solución dentro del área de la Ingeniería en Computación.

GLOSARIO

Abstracción. Es la capacidad mediante la cual una serie de objetos se categorizan en un nuevo objeto mediante una función de pertenencia. Al nuevo objeto se le denomina *clase* o *tipo de objeto*, y todos los elementos categorizados en esta clase tienen propiedades comunes, las cuales caracterizan la clase. La abstracción permite ocultar los detalles, simplificando la descripción de un problema mediante la agrupación de elementos con propiedades comunes que intervienen en el mismo³³.

Ancho de banda. Indicador de la cantidad de datos que pueden transmitirse en determinado periodo de tiempo por un canal de transmisión, por ejemplo un radiotransmisor, una antena parabólica o el cableado que conecta a dos computadoras. Por lo general, el ancho de banda se expresa en ciclos por segundo (hercios, Hz), o en bits por segundo (bps)³⁴.

API es la abreviatura de **A**plicacion **P**rogramming **I**nterface. No es más que una serie de servicios o funciones que el Sistema Operativo ofrece al programador, como por ejemplo, imprimir un carácter en pantalla, leer el teclado, escribir en un fichero de disco, etc. Visto desde la perspectiva del código máquina, el API aparece como una serie de llamadas (en otros sistemas operativos se hace mediante saltos a supervisor; en OS/2 se implementan como *Far Calls*), mientras que si lo vemos desde la de un lenguaje de alto nivel, el API aparece como un conjunto de procedimientos y funciones³⁵.

Applet. Programa dinámico e interactivo que se puede ejecutar dentro de una página Web, desplegada por un visualizador con capacidad Java³⁶.

Bases de datos analíticas. Éstas son bases de datos de sólo lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones y tomar decisiones³⁷.

Bases de datos de red. Éste es un modelo ligeramente distinto del jerárquico, en donde su diferencia fundamental es la modificación del concepto de un *nodo*, permitiendo que un mismo nodo tenga varios padres (algo no permitido en el modelo jerárquico). Fue una gran mejora con respecto al modelo jerárquico, ya que ofrecía una solución eficiente al problema de redundancia de datos, pero aun así, la dificultad que significa administrar la información en una base de datos de red, ha significado que sea un modelo utilizado en su mayoría por programadores más que por usuarios finales³⁸.

Base de datos distribuida (BDD). Conjunto de múltiples bases de datos *lógicamente relacionadas* las cuales se encuentran distribuidas entre diferentes sitios interconectados por una red de comunicaciones³⁹.

Bases de datos jerárquicas. Éstas son bases de datos que, como su nombre indica, almacenan su información en una estructura jerárquica. En este modelo los datos se organizan en una forma similar a un árbol (visto al revés), en donde un *nodo padre* de información puede tener varios

³³ Luque Ruíz, Irene. Gómez-Nieto, Miguel Angel. "Bases de Datos desde Chen hasta Codd con Oracle". Alfa Omega Rama. 2002

³⁴ <http://www.terra.es/personal/lermon/esp/enciclo.htm>

³⁵ Arboles Sergio, Navarro Luis. "Visual Basic 6 a fondo". España. Inforbook's, sl. 2002. 831p.

³⁶ Lemay, Laura; Perkins, Charles L. "Aprendiendo Java en 21 días." Prentice Hall. México 1996.

³⁷ http://es.wikipedia.org/wiki/Computaci%C3%B3n_distribuida.

³⁸ *Ibid.*

³⁹ <http://www.glosarium.com/term/1112,14,xhtml>

hijos. El nodo que no tiene padres se le conoce como *raíz*, y a los nodos que no tienen hijos se les conoce como *hojas*. Una de las principales limitaciones de este modelo, es su incapacidad de representar eficientemente la redundancia de datos⁴⁰.

Bases de datos operacionales. Éstas son bases de datos más dinámicas, orientadas a almacenar información que es modificada con el tiempo, permitiendo operaciones como actualización y adición de datos, además de las operaciones fundamentales de consulta⁴¹.

Bases de datos relacionales. Éste es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. Tras ser postuladas su base en 1970 por Edgar Frank Codd, de los laboratorios IBM en San José (California), no tardó en consolidarse como un nuevo paradigma en los modelos de base de datos. Su idea fundamental es el uso de "tablas", compuestas de *registros* (las filas de una tabla) y *campos* (las columnas de una tabla). En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia (a diferencia de otros modelos como el jerárquico y el de red). Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario casual de la base de datos. La información puede ser recuperada o almacenada por medio de "consultas" que ofrecen una amplia flexibilidad y poder para administrar la información⁴².

Bytecodes. Conjunto de instrucciones muy parecidas al código máquina, pero no son específicas para un procesador⁴³.

Cardinalidad. Número de tuplas de una relación⁴⁴.

CGI (Common Gateway Interface, Interface de Entrada Común) es un programa que se ejecuta en tiempo real en un Web Server (Servidor de Internet) en respuesta a una solicitud de un Browser (buscador). Cuando esto sucede el Web Server ejecuta un proceso hijo que recibirá los datos que envía el usuario (en caso de que los haya), pone a disposición del mismo algunos datos en forma de variables de ambiente y captura la salida del programa para enviarlo como respuesta al Browser⁴⁵.

Clase. Descripción de un conjunto de objetos; consta de atributos y métodos que resumen características comunes de un conjunto de objetos⁴⁶.

Compilador. Programa que lee un programa escrito en un lenguaje, el lenguaje fuente, y lo traduce a un programa equivalente en otro lenguaje, el lenguaje objeto⁴⁷.

Computación distribuida. Es un nuevo modelo para resolver problemas de computación masiva utilizando un gran número de computadoras organizadas en racimo incrustadas en una infraestructura de telecomunicaciones distribuida. La computación distribuida ha sido diseñada para resolver problemas demasiado grandes por cualquier simple supercomputadora, mientras mantiene la flexibilidad de trabajar en múltiples problemas más pequeños. Por tanto, la computación distribuida es un entorno multi-usuario. Debido a esta razón, las técnicas de

⁴⁰ http://es.wikipedia.org/wiki/Computaci%C3n_distribuida.

⁴¹ *Ibid*,

⁴² *Ibid*.

⁴³ Lemay, Laura; Perkins, Charles L. Op cit., p.83

⁴⁴ Mejía Argueta, Miguel Angel. "Apuntes: Introducción al Diseño de Bases de Datos." México. Marzo 2004. Dirección General de Servicios de Cómputo Académico. 55 p.

⁴⁵ <http://www.terra.es/personal/lermon/esp/enciclo.htm>

⁴⁶ Arboles Sergio, Navarro Luis. Op cit., p. 83.

⁴⁷ *Ibid*.

autorización segura son esenciales para permitir que los recursos informáticos sean controlados por usuarios remotos (distantes)⁴⁸.

Consistencia. El concepto de consistencia nos lleva necesariamente al de integridad referencial. Desde luego, una base de datos que se halle en estado de inconsistencia puede suministrar información incorrecta o contradictoria⁴⁹.

DBA (Data Base Administration). Persona que tiene el control centralizado sobre el sistema de base de datos y que controla tanto los datos como los programas que tienen acceso a ellos⁵⁰.

Funciones

- Decidir el contenido de la base de datos
- Crear la estructura de almacenamiento y los métodos de acceso
- Modificar la base de datos o la descripción de la organización física
- Otorgar permisos de acceso y prioridades a los diferentes usuarios
- Especificar las limitaciones de integridad
- Ser el enlace con los usuarios
- Definir estrategias para respaldo y recuperación.

Default. Indica el valor que tendrán ciertos datos en caso de dos o más tablas diferentes. Una vista puede ser tratada como una tabla para su manejo. Resultado de una consulta o join entre dos tablas. Ventajas: los mismos datos pueden ser vistos por diferentes usuarios. Los usuarios pueden protegerse contra cambios en la estructura de las tablas⁵¹.

Diagrama de flujo. Diagrama secuencial empleado en muchos campos para mostrar los procedimientos detallados que se deben seguir al realizar una tarea, como un proceso de fabricación. También se utilizan en la resolución de problemas, como por ejemplo en algoritmos. Los diagramas de flujo se usan normalmente para diseñar programas de computadoras⁵².

Diccionario de datos. Base de datos acerca de la terminología que se utilizará en un sistema de información. Para comprender mejor el significado de un diccionario de datos, puede considerarse su contenido como "datos acerca de los datos"; es decir, descripciones de todos los demás objetos (archivos, programas, informes, sinónimos...) existentes en el sistema. Un diccionario de datos almacena la totalidad de los diversos esquemas y especificaciones de archivos, así como sus ubicaciones. Si es completo incluye también información acerca de qué programas utilizan qué datos, y qué usuarios están interesados en unos u otros informes. Por lo general, el diccionario de datos está integrado en el sistema de información que describe⁵³.

Dominio. Conjunto de todos los valores posibles para un atributo⁵⁴.

Gateway. Conjunto de *hardware* y *software* que conecta redes que utilizan protocolos de comunicación diferentes, o que transmite datos por una red entre dos aplicaciones no compatibles. El *gateway* cambia el formato de los datos de manera que los pueda entender la aplicación que los recibe. El término se suele usar para describir cualquier computadora que transmite datos de una red a otra, pero esta acepción, técnicamente, no es correcta⁵⁵.

Grado de una tupla. Número de atributos que tiene una tupla⁵⁶.

⁴⁸ http://es.wikipedia.org/wiki/Computaci%F3n_distribuida.

⁴⁹ Mejía Argueta, Miguel Ángel. Op cit., p. 84

⁵⁰ Ibid.

⁵¹ Ibid.

⁵² <http://www.terra.es/personal/lermon/esp/enciclo.htm>

⁵³ Ibid.

⁵⁴ Mejía Argueta, Miguel Ángel. Op cit., p. 84

⁵⁵ Ibid.

⁵⁶ Ibid.

Herencia. Permite crear muchas clases que son similares entre sí, sin tener que describir cada vez las partes que son similares; esta propiedad permite combinar varias clases en una de ellas o modificar una clase existente sin modificar realmente el código original⁵⁷.

Indexación. Se refiere a la incorporación de un sitio web a los índices de un Motor de Búsqueda. La Indexación es llevada a cabo por los Motores de Búsqueda a través de sus Spiders (robots), los cuales analizan la 'tela de araña' que conforman todos los sitios de Internet. Un sitio web bien planeado hará que ésta Indexación sea realizada en poco tiempo, y logrará que los algoritmos de los Motores de Búsqueda favorezcan a éste en la posición dentro de sus índices⁵⁸.

Índices. Los índices mejoran el desempeño de las consultas. Pueden ser: Non Clustered (0-250, utilizan apuntadores) y Clustered (0-1, hacen una copia física de la tabla, la cual utiliza el 120% de la tabla)⁵⁹.

Integridad. Se da cuando en una relación de información se modifica algún elemento que se encuentre en varias tablas sin afectar su contenido, ya que cada uno de ellos puede poseer información diferente⁶⁰.

Interfaz. Punto en el que se establece una conexión entre dos elementos, que les permite trabajar juntos. En el campo de la informática se distinguen diversos tipos de interfaces que actúan a diversos niveles, desde las interfaces claramente visibles, que permiten a las personas comunicarse con los programas, hasta las imprescindibles interfaces *hardware*, a menudo invisibles, que conectan entre sí los dispositivos y componentes dentro de las computadoras. Las interfaces de usuario cuentan con el diseño gráfico, los comandos, mensajes y otros elementos que permiten a un usuario comunicarse con un programa⁶¹.

Interfaz de programación de aplicaciones. Conjunto de rutinas que utiliza un programa de aplicación para solicitar y efectuar servicios de nivel inferior ejecutados por un sistema operativo informático. Un programa de aplicación efectúa dos tipos de tareas: las relacionadas con el trabajo que se está realizando, por ejemplo aceptar la entrada de texto o de números en un documento u hoja de cálculo, y las relacionadas con las tareas de mantenimiento, como la gestión de archivos y la presentación de la información en la pantalla. Estas tareas de mantenimiento son realizadas por el sistema operativo y la interfaz de programación de aplicaciones (API) proporciona al programa los medios para comunicarse con el sistema, indicándole qué tarea básica del sistema debe realizar y cuándo⁶².

Interfaz gráfica de usuario. Tipo de visualización que permite al usuario elegir comandos, iniciar programas y ver listas de archivos y otras opciones utilizando las representaciones visuales (iconos) y las listas de elementos del menú. Las selecciones pueden activarse bien a través del teclado o con el ratón⁶³.

Internet. Es una red de computadoras interconectadas entre sí que ofrecen acceso y comparten información a través de un lenguaje común. En la actualidad es la red de computadoras más grande que existe en el mundo; se conecta por teléfono (a través de un módem) o por fibra óptica y transmite toda clase de información⁶⁴.

⁵⁷ Arboles Sergio, Navarro Luis. Op cit., p.84

⁵⁸ <http://www.e-ingenieros.cl/conceptos/indexacion.htm>

⁵⁹ Mejía Argueta, Miguel Angel. Op cit., p.85

⁶⁰ Ibid.

⁶¹ <http://www.terra.es/personal/lermon/esp/enciclo.htm>

⁶² Ibid.

⁶³ Ibid.

⁶⁴ Trejos, Hermanos Sucesores. "Aprenda redes visualmente". IDG Books. 1997. 1ª Edición.

Interprete. En lugar de producir un programa objeto como resultado de una traducción, realiza las operaciones que implica el programa fuente. Muchas veces los intérpretes se usan para ejecutar lenguajes de órdenes, pues cada operador que se ejecuta en un lenguaje de órdenes suele ser una invocación de una rutina compleja⁶⁵.

Intranet. Es un site (lugar) privado o un conjunto de sites configurados para el uso exclusivo de una organización. Es una versión cerrada que sólo conecta al usuario con computadoras seleccionadas⁶⁶.

Lenguaje de Programación. Es aquel que utilizan los programadores para escribir un programa en forma más o menos cómoda y que por lo general requiere de una traducción (ensamble, compilación, transcripción) para ser transformado a lenguaje de máquina⁶⁷.

Llave candidata. Atributo o conjunto de atributos que podrían servir como llaves primarias⁶⁸.

Llave extranjera o foránea. Son llaves que son compartidas por dos tablas para lograr una relación entre ellas⁶⁹.

Llave primaria. El atributo que identifica de manera única a un registro⁷⁰.

Llave secundaria. Son llaves que tienen todas las características para ser primarias, pero que por una razón o por otra no fueron tomadas como tales, ya que hubo otra (s) que cumplían mejor con ese objetivo⁷¹.

Mainframe. Computadora de alta capacidad diseñada para las tareas computacionales más intensas. Las computadoras de tipo mainframe suelen tener varios usuarios, conectados al sistema a través de terminales. Los mainframes más potentes, llamados supercomputadoras, realizan cálculos muy complejos y que requieren mucho tiempo. Este tipo de equipos informáticos lo utilizan principalmente los científicos dedicados a la investigación pura y aplicada, las grandes compañías y el ejército⁷².

Modelo de Datos. Mediante el uso de un modelo de datos se describen las propiedades que caracterizan el fenómeno y que lo diferencian de otros fenómenos que se puedan o no describir, las relaciones entre estas propiedades, y cómo las propiedades y las relaciones pueden evolucionar con el tiempo. Mediante un modelo de datos se describen las características estáticas y dinámicas de un fenómeno. No proporciona una interpretación completa de cómo puede utilizarse la información descrita por el mismo, sino únicamente a qué operaciones o a qué tratamiento puede ser sometida esta información⁷³.

MVCC (Multi-Version Concurrency Control, Control de Concurrencia Multi-Versión). Es una técnica avanzada para mejorar las prestaciones de una base de datos en un entorno multiusuario⁷⁴.

Motores de Búsqueda. Son empresas en Internet, dedicadas a crear índices de sitios web. Para hacer esta indexación, los Motores de Búsqueda hacen correr a través de la Red Mundial de

⁶⁵ Aho, Alfredo., SEIT Ravi. Addison Wesley Iberoamericana. 1999.

⁶⁶ Trejos, Hermanos Sucesores. Op cit., 87

⁶⁷ Molino, Enzo, Mora, José Luis. "Introducción a la Informática". Trillas. 1982. 1ª Edición.

⁶⁸ Mejía Argueta, Miguel Angel. Op cit., p.85

⁶⁹ Ibid.

⁷⁰ Ibid.

⁷¹ Ibid.

⁷² <http://www.terra.es/personal/lrmon/esp/enciclo.htm>

⁷³ Luque Ruíz, Irene. Gómez-Nieto, Miguel Angel. Op cit., p. 83

⁷⁴ <http://es.tldp.org/Postgresqt-es/web/navegable/user/mvcc.html>

Internet, un programa automático que permite ir recorriendo los sitios web y al mismo tiempo los va ubicando en un índice. A estos 'espías electrónicos' se les llama Spiders⁷⁵.

Nivel Conceptual. A este nivel son representados los tipos o clases de objetos, y sus relaciones desde un punto de vista estructural. En el nivel conceptual se representa un modelo del sistema en el que se describen cada uno de los tipos de objetos o elementos del mismo. Para cada uno de estos tipos de objetos se describen sus propiedades y el dominio o tipo de datos básico en el cual pueden ser medidas, así como las restricciones o límites de los valores que pueden representarse para cada una de estas propiedades. Además, son descritas las relaciones entre los tipos de objetos, relaciones jerárquicas o no. Se representa el problema tal y como es; es decir, se representa el mundo real del problema tal y como se percibe, sin tener en cuenta cómo este problema puede ser representado para que sea entendido por los programas de computación⁷⁶.

Nivel de Vistas. El nivel más alto de abstracción describe sólo parte de la base de datos completa. A pesar del uso de estructuras más simples en el nivel lógico, queda algo de complejidad, debido al gran tamaño de la base de datos⁷⁷.

Nivel Físico. El nivel más bajo de abstracción describe cómo se almacenan realmente los datos. En el nivel físico se describen en detalle las estructuras de datos complejas de bajo nivel. El principio de representación del problema está guiado tanto por el soporte utilizado para su representación como por los métodos o mecanismos que se van a utilizar para el tratamiento de la información correspondiente a éste. En este nivel, el problema se representa en la forma en que es visto por el sistema de representación y tratamiento utilizado, y no como existe o es visto desde el mundo real⁷⁸.

Nivel Lógico. El siguiente nivel más alta de abstracción describe *qué* datos se almacenan en la base de datos y *qué* relaciones existen entre esos datos. Se representa el problema bajo las limitaciones impuestas por la representación y el tratamiento de la información que se vaya a realizar. Es decir, en este punto se introducen en la representación las limitaciones o restricciones que imponen los mecanismos o soportes que se van a utilizar para la representación y tratamiento de la información del problema⁷⁹.

OLTP (On-line Transaction Processing, Proceso de la Transacción en-línea). Es un tipo de proceso especialmente rápido en el que las solicitudes de los usuarios son resueltas de inmediato; naturalmente, ello implica la concurrencia de un «mecanismo» que permite el procesamiento de varias transacciones a la vez⁸⁰.

Páginas Web. Son documentos en la WWW. Estas páginas pueden incluir texto, ilustraciones, sonido, video y animación⁸¹.

Paradigma. Conjunto de formas que sirven de modelo en los diversos tipos de flexión⁸².

Procedimientos almacenados. Programas SQL que podemos llamar por su nombre (son ejecutables)⁸³.

Proceso distribuido. Método de procesamiento de la información en el que varios procesos (programas en ejecución) en paralelo, en la misma máquina o distribuidos entre computadoras

⁷⁵ http://www.e-ingenieros.cl/conceptos/motores_busqueda.htm

⁷⁶ Luque Ruíz, Irene. Gómez-Nieto, Miguel Angel. Op cit., p. 87

⁷⁷ Ibid.

⁷⁸ Ibid.

⁷⁹ Ibid.

⁸⁰ <http://www.glosarium.com/term/1112,14.xhtml>

⁸¹ Trejos, Hermanos Sucesores. Op cit., p. 87.

⁸² <http://www.glosarium.com/term/1112,14.xhtml>

⁸³ <http://www.ica.luz.ve/~carevalo/procesamiento-distribuido-1/c1.html>

separadas interconectadas a través de una red de comunicaciones, colaboran en la realización de una tarea. Esta colaboración puede ser tan sencilla como distribuir la carga de trabajo entre procesos idénticos, en el caso de una red de cajeros automáticos, o tan compleja como multitud de procesos distintos, interdependientes, controlando el vuelo de una nave espacial⁸⁴.

Programación orientada a objetos. Estilo de programación en el que un programa se contempla como un conjunto de objetos limitados que, a su vez, son colecciones independientes de estructuras de datos y rutinas que interactúan con otros objetos. Una clase define las estructuras de datos y rutinas de un objeto. Un objeto es una instancia de una clase, que se puede usar como una variable en un programa. En algunos lenguajes orientados a objetos, éste responde a mensajes, que son el principal medio de comunicación. En otros lenguajes orientados a objeto se conserva el mecanismo tradicional de llamadas a procedimientos⁸⁵.

Programación lineal. Técnica matemática y de investigación de operaciones que se utiliza en la planificación administrativa y económica para maximizar las funciones lineales de un gran número de variables sujetas a determinadas restricciones. El desarrollo de computadoras electrónicas y de técnicas de procesamiento de alta velocidad ha aportado recientemente muchos avances a la programación lineal, de forma que ahora esta técnica se utiliza extensamente en operaciones industriales y militares⁸⁶.

Redundancia. Repetición de los mismos datos o elementos a través de diferentes registros, aplicaciones o archivos. Esto sucede cuando en las bases de datos cada aplicación tiene sus propios archivos privados, esto provoca considerable redundancia en los datos almacenados, con el consecuente desperdicio de espacio de almacenamiento⁸⁷.

Reglas. Actúa como una máscara de edición (da una condición) para una columna o un tipo de dato definido por el usuario⁸⁸.

Relación. Conjunto que representa una correspondencia entre dos o más conjuntos, una relación es, por tanto, un nuevo conjunto en que cada elemento está formado por la agregación de los elementos de los conjuntos individuales que intervienen en la relación⁸⁹.

Seguridad. La seguridad implica asegurar que los usuarios están autorizados para llevar a cabo lo que tratan de hacer. El problema de la seguridad tiene muchos aspectos, entre ellos los siguientes⁹⁰:

- Aspectos legales, sociales y éticos.
- Controles físicos.
- Cuestiones de política interna.
- Problemas de operación.
- Controles de equipo.
- Seguridad del sistema operativo.
- Seguridad de Objetos. Se refiere a los permisos de los diferentes usuarios para poder hacer uso de tablas, procedimientos almacenados, triggers, etc.
- Seguridad de operaciones. Aquí se manejan permisos para poder modificar (insertar, borrar, actualizar) la base de datos.

⁸⁴ <http://www.terra.es/personal/l/ermon/esp/enciclo.htm>

⁸⁵ Ibid.

⁸⁶ Ibid.

⁸⁷ Mejía Argueta, Miguel Angel. Op cit., p.85

⁸⁸ Ibid.

⁸⁹ Luque Ruíz, Irene. Gómez-Nieto, Miguel Angel. Op cit., p. 87

⁹⁰ Mejía Argueta, Miguel Angel. Op cit., p.85

Servidor de archivos. Dispositivo de almacenamiento de archivos en una red de área local al que todos los usuarios de la red pueden acceder. A diferencia de un servidor de disco, que aparece ante el usuario como una unidad de disco remota, un servidor de archivos es un dispositivo más complejo que no sólo almacena archivos sino que también los administra y los mantiene en orden a medida que los usuarios de la red los solicitan y los modifican. Para gestionar las tareas de manejo de varias solicitudes (a veces simultáneas), un servidor de archivos cuenta con un procesador y *software* de control, así como una unidad de disco para el almacenamiento. En redes de área local, un servidor de archivos suele ser una computadora con un disco duro grande que está dedicado exclusivamente a las funciones de administración de archivos compartidos⁹¹.

Sistema de bases de datos distribuida (SBDD). Sistema en el cual múltiples sitios de bases de datos están ligados por un sistema de comunicaciones, de tal forma que, un usuario en cualquier sitio puede acceder los datos en cualquier parte de la red exactamente como si los datos estuvieran almacenados en su sitio propio⁹².

Sistema de manejo de bases de datos distribuidas (SMBDD). Es aquel que se encarga del manejo de la BDD y proporciona un mecanismo de acceso que hace que la distribución sea *transparente* a los usuarios. El término transparente significa que la aplicación trabajaría, desde un punto de vista lógico, como si un solo SMBD ejecutado en una sola máquina, administrara esos datos⁹³.

Sistemas distribuidos. Sistemas de cómputo que utilizan más de una computadora o procesador para ejecutar una aplicación. Esta definición incluye el procesamiento paralelo: varios CPU's ejecutan una aplicación, el procesamiento distribuido se refiere generalmente a computadoras conectadas en red⁹⁴.

Sitios Web. Es una colección de páginas Web promovida por un colegio, universidad, agencia gubernamental, compañía o individuo. Un sitio Web es una excelente forma de promover un servicio o producto a millones de personas⁹⁵.

Tabla. Formada por campos o atributos (columnas) y registros o tuplas (filas)⁹⁶.

Técnicas de cuarta generación. Abarca un amplio espectro de herramientas de software que tienen algo en común: todas facilitan al Ingeniero del software, la especificación de algunas características del software de alto nivel. Luego, la herramienta genera automáticamente el código fuente basándose en la especificación del técnico⁹⁷.

Tipos de datos. Indican la forma en que serán almacenados los datos (entero, flotante, carácter, binario, etc)⁹⁸.

- Numéricos: Numero positivos o negativos con o sin punto decimal, integer, smallint, decimal, float, long, double
- Texto: o combinaciones de texto y números que no requieran cálculos, como los números de teléfono, char, varchar, text, memo.
- Fecha/hora: Datetime.

⁹¹ <http://www.terra.es/personal/lermon/esp/enciclo.htm>

⁹² <http://www.glosarium.com/term/1112,14.xhtml>

⁹³ Ibid.

⁹⁴ <http://www.ica.luz.ve/~carevalo/procesamiento-distribuido-1/c1.html>

⁹⁵ Luque Ruíz, Irene. Gómez-Nieto, Miguel Angel. Op cit., p. 87

⁹⁶ Mejía Argueta, Miguel Angel. Op cit., p.85

⁹⁷ Lemay, Laura; Perkins, Charles L. Op cit., p.83

⁹⁸ Ibid.

Triggers. Cuidan la integridad referencial de los datos (si se quiere modificar un campo llave, primero lo guarda, luego modifica todos los datos relacionados con el campo y entonces lo modifica)⁹⁹.

Tupla. Conjunto de valores que componen un renglón de la relación. Es equivalente a una instancia de un registro¹⁰⁰.

World Wide Web. Es uno de los servicios disponibles más utilizados en la Internet actualmente. Esta consiste en una amplia colección de documentos almacenados en las computadoras alrededor del mundo¹⁰¹.

⁹⁹ Ibid.

¹⁰⁰ Mejía Argueta, Miguel Angel. Op cit., p.85

¹⁰¹ Trejos, Hermanos Sucesores. Op cit., p. 88

BIBLOGRAFÍA

LIBROS

- "Pequeño Larousse Ilustrado". México. 1982. 1662 p.
- Aho, Alfredo., SEIT Ravi. "**Compiladores. Principios, técnicas y herramientas**". Addison Wesley Iberoamericana. 1999.
- Arboles Sergio, Navarro Luis. "**Visual Basic 6 a fondo**". España. Inforbook's, sl. 2002. 831 p.
- De Miguel, Adoración y Piattini Velthus, Mario Gerardo. "**Fundamentos y Modelos de Bases de Datos**". México. 1998. Alfaomega. 515 p.
- Díaz Pablo, Galeano German. " **Access 2000**". Prentice may. 1999
- Fairley, Richard. "**Ingeniería de Software**". Addison Wesley. 1988.
- González, Martín Oscar, Ruiz González, Francisco. "**Arquitecturas de Sistemas de Bases de Datos**". Universidad de Castilla la Mancha. 1999/2000.
- Lemay, Laura; Perkins, Charles L. "**Aprendiendo Java en 21 días.**" Prentice Hall. México 1996
- López Vega, Marco Antonio **Apuntes: "Redes de Datos"** Universidad Nacional Autónoma de México.
- Luque Ruíz, Irene. Gómez-Nieto, Miguel Angel. "**Bases de Datos desde Chen hasta Codd con Oracle**". Alfa Omega Ra-Ma. 2002
- Mejía Argueta, Miguel Angel. "**Apuntes: Introducción al Diseño de Bases de Datos.**" México. Marzo 2004. Dirección General de Servicios de Cómputo Académico. 55 p.
- Molino, Enzo, Mora, José Luis. "**Introducción a la Informática**". Trillas. 1982. 1ª Edición.
- Presuman, Roger S. "**Ingeniería del Software. Un enfoque práctico**". Mc Graw Hill. 1989. 4ª Edición.
- Raya, José Luis. "**Redes Locales y TCP/IP**". Computec.
- Silberschatz, Abraham, F. Kort, Henry. "**Fundamentos de Bases de Datos**". Mc Graw-Gill. 1998.
- Sommerville, Ian. "**Ingeniería de Software**". Addison Wesley. 2002. 6ª Edición.
- Trejos, Hermanos Sucesores. "**Aprenda redes visualmente**". IDG Books. 1997. 1ª Edición.

MESOGRAFÍA

- En la siguiente dirección se encuentra una descripción del modelo Cliente/Servidor.

<http://www.arcride.edu.ar/appei/revistas/reva2n11a2.htm>

- En las siguientes direcciones se encuentra información sobre el Sistema Gestor de Bases de Datos Postgresql:

<http://advocacy.postgresql.org/advantages/?lang=es>

www.postgresql.com

- En esta dirección se encuentran algunas ventajas y desventajas de los lenguajes de programación Java y Visual Basic:

<http://www.qualitrain.com.mx/objeIndirecto/javavsvbasic.htm>

- En esta dirección se encuentra algunas características del Sistema Gestor de Bases de Datos Access:

http://www.microsoft.com/spain/office/access/access_5.asp

- En esta dirección se encuentran algunos términos referentes a Bases de Datos:

<http://www.jegsworks.com/Lessons-sp/lesson1-2/lesson2-4database.htm>

www.monografias.com

- En estas direcciones se encuentra información referente a las redes de computadoras:

<http://www.linti.unlp.edu.ar/trabajos/tesisDeGrado/tutorial/redes/tipredes.htm>

<http://iio.ens.uabc.mx/jmilanez/escolar/redes/01110100.html>

<http://iio.ens.uabc.mx/jmilanez/escolar/redes/03020000.html>

<http://www.vanco.es/ContentManager/Document.asp?CombinedId=5D306D504>

- En estas direcciones se encuentran diversos conceptos referentes al desarrollo de este trabajo, tales como Bases de Datos, tipos de redes de computadoras, etc.:

<http://www.terra.es/personal/lermon/esp/enciclo.htm>

<http://www.glosarium.com/term/1112,14,xhtml>

<http://es.tldp.org/Postgresql-es/web/navegable/user/mvcc.html>

http://es.wikipedia.org/wiki/Computaci%F3n_distribuida

<http://www.ica.luz.ve/~carevalo/procesamiento-distribuido-1/c1.html>

<http://www.e-ingenieros.cl/conceptos/indexacion.htm>

http://www.e-ingenieros.cl/conceptos/motores_busqueda.htm