UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

INFORME DE ACTIVIDADES PROFESIONALES

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACION

P R E S E N T A :

ANAID VICTORIA VELÁZQUEZ RIVERA

AVAL: L.I. LUZ MARIA RAMÍREZ ROMERO

CIUDAD UNIVERSITARIA 2006

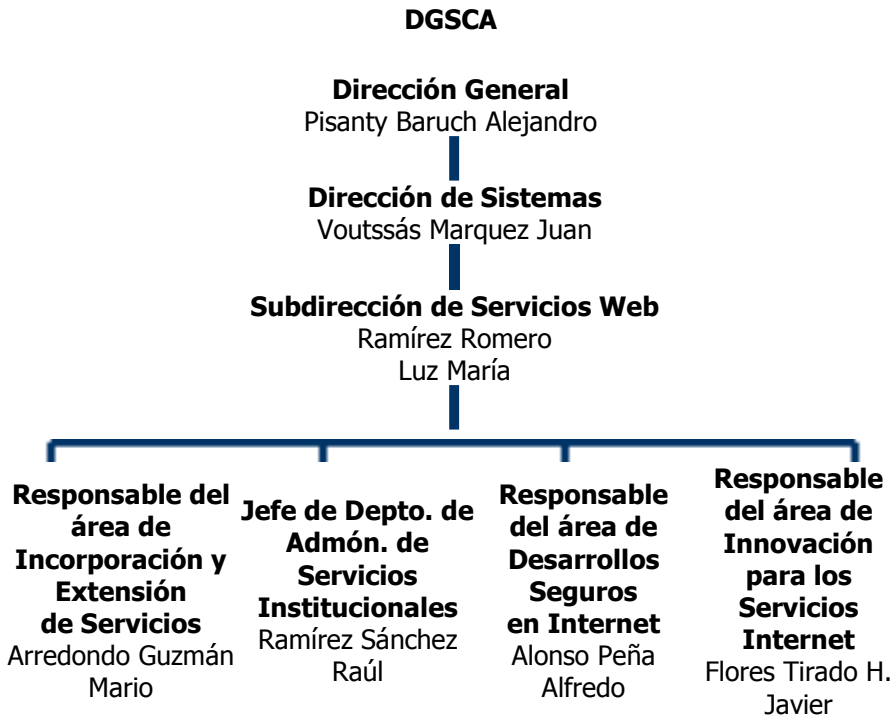
INFORME DE ACTIVIDADES PROFESIONALES

INTRODUCCIÓN

El siguiente trabajo describe las actividades profesionales en el área de la Ingeniería en Computación, las cuales fueron realizadas en la Subdirección de Servicios Web de la DGSCA, UNAM, a cargo de la subdirectora L.I. Luz María Ramírez Romero.

La Subdirección de Servicios Web SSW tiene como objetivo principal, desarrollar e implementar sitios y/o sistemas sobre Internet para: dependencias de la UNAM, instituciones privadas y gubernamentales.

El equipo de trabajo que integra la SSW es multidisciplinario y aunque yo pertenecí al área de Incorporación y Extensión de Servicios trabajé conjuntamente con las demás áreas durante el desarrollo de los sistemas. El siguiente diagrama muestra la organización de las áreas en la Subdirección de Servicios Web. [1]



[1] El diagrama fue tomado de la página de la Subdirección de Servicios Web http://www.serviciosweb.unam.mx/acerca.html#organizacion

Departamento de Administración de Servicios Institucionales[2]

Objetivos:

- Crear el concepto y la imagen gráfica de los sitios y portales Internet que lo solicitan a la Subdirección.
- Organizar la información del usuario y diseñar el sistema de navegación adecuado al grupo de usuarios objetivo de un Portal o sitio Internet.
- Desarrollar presentaciones e imágenes gráficas dinámicas para proporcionar al usuario una propuesta diferente, cuando su sitio así lo requiera.
- Incorporar nuevos servicios de Internet a los sitios desarrollados por el departamento.
- Incrementar la satisfacción del cliente/usuario brindando un mejor servicio y asistencia personalizada.

Área de Incorporación y Extensión de Servicios

Objetivo:

- Analizar, diseñar e implementar portales, sitios web dinámicos, sistemas para los usuarios, y servicios web de calidad, de acuerdo a normas y estándares internacionales, para usuarios universitarios y externos a la UNAM.

Área de Innovación para los Servicios Web.

Objetivo:

- Desarrollo de prototipos y servicios web de calidad, utilizando nuevas metodologías, estándares, tecnologías y herramientas para mantener a la vanguardia el desarrollo de servicios y sistemas de Internet.

Área de Desarrollos Seguros en Web.

Objetivos:

- Desarrollar aplicaciones web eficientes y robustas para dependencias universitarias e instituciones externas a la UNAM en base a estándares de seguridad.
- Dotar de elementos de seguridad a los desarrollos y a la red interna de la Subdirección.

Cada área está integrada por alumnos de la UNAM y otras universidades, egresados y profesionales relacionados al área de cómputo, quienes se incorporan en un ambiente de trabajo, el cual les brinde capacitación, cursos y práctica en diversos temas relacionados al cómputo. Las actividades son realizadas por medio de servicio social, prácticas profesionales, programa de becarios y contrataciones por honorarios.

Hoy en día uno de los principales medios de comunicación para presentar la información es la Internet, por ello los sistemas sobre Internet son desarrollados para optimizar tiempo, distancia y recursos físicos.

[2] Información tomada de la página de la Subdirección de Servicios Web
http://www.serviciosweb.unam.mx/acerca.html#organizacion

Las tecnologías utilizadas en el desarrollo de los sistemas sobre Internet varían de acuerdo a los requerimientos de los sistemas, algunas tecnologías utilizadas dentro de mi periodo en el ámbito laboral fueron:
Lenguaje de presentación estándar para páginas Internet: HTML
Lenguajes de programación: JavaScript, PHP, Visual Basic.Net
Manejadores de bases de datos: Mysql, Postgres SQL, SQL Server.

Metodología para el desarrollo de sistemas de información

Ingeniería de software para el desarrollo de sistemas

La ingeniería de software es una disciplina tecnológica y administrativa que se relaciona con la producción y mantenimiento sistemático de productos de software que se desarrollan, modifican a tiempo considerando sus estimaciones de costos. Un producto de software tiene múltiples encargados de su desarrollo, usuarios y responsables de su mantenimiento[1]. Los objetivos principales de la ingeniería de software son mejorar la calidad de los productos e incrementar la productividad y satisfacción en el trabajo

Un producto de software consiste en varios artículos para su entrega, además del código fuente, manual de usuario, informe de verificación del software, conjunto de pruebas, guía de mantenimiento, instrucciones de instalación y materiales para entrenamiento. También requiere de varios documentos como son: definición del producto, plan de control de calidad, especificaciones de los requisitos del software, contrato con el cliente, minutas de los recorridos, inspecciones y encuentros para revisión del proyecto y documento del legado del proyecto.

La conducción de las actividades del proyecto, dentro de un marco de trabajo por fases de marcas de logros y productos de trabajo estándar, garantiza un proceso de desarrollo ordenado.

a) Definición del producto

Los principales componentes de una definición del producto son: Una descripción concisa del problema que se solucionará, una propuesta de objetivos del sistema y proyecto, identificación de las características del usuario, funciones que el sistema realizará, estrategia de solución, prioridades para las características del producto y los criterios para la aceptación de éste.

b) Plan del proyecto

Sus componentes incluyen: modelo del ciclo de vida que se usará, la estructura de la organización, las estimaciones preliminares de requerimientos de personal y recursos, estimación del costo preliminar, calendario de desarrollo preliminar y las especificaciones para el monitoreo del proyecto y los mecanismos de control.

[1] Richard E. Fairley, Ingeniería de Software, Capitulo 10, pág. 335

El modelo del ciclo de vida define la terminología, las marcas de logros y los productos de trabajo para el proyecto. La revisión de factibilidad del proyecto es la revisión de logros para la definición del producto y el plan del proyecto. El resultado puede ser la terminación del proyecto, su aprobación o más estudio por parte del grupo de planeación.

c) Especificación de los requisitos del software

El objetivo de la definición de requisitos es producir una especificación completa y consistente de los requisitos técnicos para un producto de software, mediante notaciones formales cuando resulte apropiado.

Los componentes esenciales de una especificación de requisitos del software son una breve visión y un resumen del producto, la especificación de las interfaces externas y los flujos de datos, los requisitos funcionales, los requisitos de desempeño, manejo de las excepciones y criterios de aceptación para el producto de software.

d) Diseño del proyecto

Las actividades en esta etapa son el diseño externo, estructural y detallado. El enfoque del flujo de datos al diseño de software (diseño estructurado) ve a la conversión de los diagramas de flujo de los datos dentro de las cartas de estructura, como un asunto principal del diseñador. El enfoque de la estructura de los datos ve a la estructura de los archivos de entrada y salida, así como a la derivación de la estructura del programa basada en las estructuras de los archivos.

Saber cuándo el diseño es suficientemente detallado y cuándo empezar a convertir el diseño en una implementación es por regla práctica, considerar que un diseño está completo si los programadores están familiarizados con el lenguaje de implementación, aunque no lo estén con los requisitos del usuario, y se puede implantar el sistema trabajando solo a partir de las especificaciones de diseño. Hay dos revisiones de logros durante el diseño: La preliminar del diseño y la revisión crítica del mismo. La primera revisión se efectúa cerca del final del diseño estructural y antes del diseño detallado, mientras que la revisión crítica se realiza casi a la terminación del diseño detallado y antes de la implementación. En la revisión preliminar se verifican los requisitos del proyecto, el manual preliminar del usuario y las especificaciones del diseño estructural y externo.

e) Implementación del proyecto

El objetivo principal de la implementación del proyecto es el desarrollo de código fuente, sencillo para su lectura y comprensión. Su claridad facilita la depuración, pruebas y modificación de un producto de software. Dada una documentación adecuada del diseño, la implementación de un producto de software debe ser un proceso sencillo, de poca tensión y muy eficiente.

El requisito básico de la codificación estructurada es el uso de construcciones de una sola entrada y una sola salida. En algunos casos, este tipo de programas requerirán de segmentos de código repetidos o llamadas repetidas a las subrutinas. El apego estricto a una sola entrada y una sola salida impediría las salidas prematuras de los ciclos y las ramificaciones hacia el código para manejo de excepciones.

El respeto a los estándares y principios generales de la implementación por parte de todos los programadores de un proyecto da como resultado un producto de calidad uniforme. Se definen como estándares a aquellos aspectos que pueden ser revisados por una herramienta automatizada, mientras que el apego a un principio general requiere de interpretación humana. La marca de logro principal para la implementación del producto es la integración con éxito de los componentes del código fuente en un sistema que funcione.

f) Control de calidad

La verificación del ciclo de vida es el proceso de determinar el grado con el cual los productos de trabajo de una fase dada del ciclo de desarrollo cumplen las especificaciones de las fases previas. La verificación formal es una demostración matemática rigurosa de que el código fuente cumple con los requisitos. La validación tiene que ver con la evaluación de un producto de software al final del proceso de desarrollo para determinar su apego a los requisitos del producto. Los objetivos principales de las actividades de verificación y validación son valorar y mejorar la calidad de los productos de trabajo intermedios y los artículos por entregar un proyecto de software.

La alta calidad se logra por medio de la verificación continua de los productos de trabajo conforme se desarrollan. Los recorridos y las inspecciones pueden también ser muy útiles, estos se deben usar como oportunidades para detectar errores sin culpar a nadie.

g) Pruebas del proyecto

Las pruebas de función y de desempeño se desarrollan durante las fases de diseño, basándose en la especificación de requisitos; mientras que las de tensión se proyectan de manera intencional para sobrecargar o romper el sistema. Las pruebas de estructura se relacionan con el examen de la estructura lógica del código fuente.

Se recalca la necesidad de análisis y diseño sistemático, así como de la verificación continua de los productos de trabajo, de modo que se eliminen los errores antes de la implementación, es deseable ejecutar varias pruebas para lograr mayor confianza en el producto.

Probar es el proceso de ejecutar casos de verificación prueba con la intención de exponer errores. La depuración es el proceso de localizar y corregir la causa de un error conocido. La depuración eficiente requiere de habilidades altamente desarrolladas para la solución de problemas.

h) Mantenimiento del proyecto

El mantenimiento de software es el procedimiento usual para denotar las distintas actividades de verificación sucedidas después de la liberación del producto. Las modificaciones se realizan para mejorar, adaptar y corregir errores en los productos de software. El mantenimiento es una pequeña parte, en la cual los cambios del producto pueden implicar reanalizar, rediseñar, reimplementar y actualizar los documentos de apoyo.

La administración de la configuración es un aspecto siempre presente a lo largo del desarrollo y mantenimiento del ciclo de vida de un producto de software. Los cambios subsecuentes a los productos de trabajo requieren de un mecanismo formal de modificación. La administración de la configuración es muy importante en esta fase, debido a que la mayor parte de los productos de software se distribuyen y existen múltiples versiones y liberaciones. Su calidad de un producto de software a través de ciclos sucesivos de modificaciones y actualizaciones, es un aspecto fundamentalmente importante durante el desarrollo del software. La calidad de un producto de software se puede degradar con prontitud debido a retoques y arreglos rápidos que:

1. No garantizan el estilo de codificación
2. No mantienen los estándares de documentación
3. No actualizan los documentos de apoyo.

Metodología para el desarrollo de sistemas

Los sistemas de información basados en computadora sirven para diversas finalidades que van desde el procedimiento de las transacciones de una empresa, hasta proveer la información necesaria para decidir sobre asuntos que se presentan con frecuencia. En algunos casos los factores que deben considerarse en un proyecto de sistemas de información (tales como el aspecto más apropiado de la computadora o la tecnología de comunicaciones que se va a utilizar, el impacto del nuevo sistema sobre los empleados de la empresa y las características específicas que el sistema debe tener), se pueden determinar de una manera secuencial. En otros casos, debe ganarse experiencia por medio de pruebas conforme el sistema evoluciona por etapas.

Los métodos han evolucionado, como lo han sido los equipos de cómputo y el avance tecnológico debe ser exhaustivo en el desarrollo de sistemas. Los métodos más aplicados son:

1. Método del ciclo de vida para el desarrollo de sistemas
2. Método del desarrollo del análisis estructurado.
3. Método del prototipo de sistemas.

Método del ciclo de vida

El desarrollo de sistemas, un proceso formado por diferentes etapas, comienza cuando la administración o algunos miembros del personal responsable de desarrollar sistemas, detectan un sistema de la empresa que necesita mejoras o en su defecto, desarrollar uno que satisfaga las necesidades de la misma.

El método del ciclo de vida para el desarrollo de sistemas es el conjunto de actividades que los analistas, diseñadores y usuarios realizan para desarrollar e implantar un sistema de información. En la mayor parte de las situaciones dentro de una empresa todas las actividades están muy relacionadas, en general son inseparables y quizá sea difícil determinar el orden de los pasos que se siguen para efectuarlas.

Una idea básica del ciclo de vida para el desarrollo de sistemas es que es un proceso bien definido en el cual se percibe una aplicación y se desarrolla. Sus fases proveen una base para la administración y el control, por lo cual se definen los segmentos del flujo de trabajo que se pueden identificar para propósitos administrativos y especificar los documentos u otros resultados que van a ser producidos en cada fase.

Las fases en este ciclo para el desarrollo de sistemas de información son descritas por varios autores, pero las diferencias radican principalmente en la cantidad de detalle y en la forma de categorización. En general, el método consta de las siguientes actividades:

1. Investigación preliminar
2. Determinación de requerimientos
3. Diseño del sistema
4. Desarrollo del software
5. Prueba del sistema
6. Implementación y evaluación

1) Investigación preliminar

La solicitud para recibir apoyo de un sistema puede originarse por varias razones; sin importar cuáles sean éstas, el proceso se inicia siempre con la petición de una persona. Cuando se formula la solicitud comienza la primera actividad del ciclo: la investigación preliminar. Esta actividad, a su vez, se divide en tres partes:

a) Aclaración de la solicitud

Muchas solicitudes no están formuladas de manera clara, por consiguiente, antes de considerar cualquier investigación de sistemas, la solicitud de proyecto debe examinarse para determinar con precisión lo que el solicitante requiere.

b) Estudio de factibilidad

Cuando se propone una nueva aplicación, normalmente se lleva acabo el estudio de factibilidad antes que sea aprobada para su desarrollo, lo que da como resultado el reconocimiento tanto de los beneficios como de los riesgos inherentes al desarrollo y a la implementación del sistema de aplicación propuesto.

c) Aprobación de la solicitud

No todos los proyectos solicitados son deseables o factibles. Algunas organizaciones reciben tantas solicitudes que sólo es posible atender unas cuantas. Sin embargo, aquellos proyectos que son deseables y factibles deben incorporarse en los planes.

2) Determinación de los requerimientos del sistema

Con el fin de obtener de manera efectiva un conjunto completo y correcto de los requerimientos, es necesario utilizar un método o métodos que tengan en cuenta hasta que punto los requerimientos son completamente conocidos o por el contrario, si necesitan ser explorados o descubiertos. El resultado del análisis de los requerimientos de información del ciclo de vida de desarrollo es un reporte que detalla lo necesario para su aplicación.

Esto último consta de:

a) Informe (incluye los datos elementales en los reportes)
b) Consultas
c) Esquema conceptual de la base de datos (a partir del modelo de los datos u otro análisis)
d) Requerimientos funcionales
e) Requerimientos de interfaz de usuario

3) Diseño del sistema

El diseño del sistema produce detalles que establecen la forma en la que el sistema cumplirá con los requerimientos identificados durante la fase de análisis. Se comienza el proceso de diseño identificando los reportes y demás salidas que debe producir el sistema. Hecho lo anterior, se determinan con toda precisión los datos específicos para cada reporte y salida, siendo común que los diseñadores hagan un bosquejo del formato o pantalla que esperan aparezca cuando el sistema esté finalizado.

El diseño de un sistema también indica los datos de entrada, aquellos que serán calculados y los que deben ser almacenados. Asimismo, se escriben con todo detalle los procedimientos de cálculo y los datos individuales. Los diseñadores seleccionan las estructuras de archivo y dispositivos de almacenamiento. Los procedimientos que se escriben indican cómo procesar datos y producir salidas.

Los documentos que contienen las especificaciones de diseño representan a éste de muchas maneras (diagramas, tablas y símbolos especiales). La información detallada del diseño proporciona al equipo de programación, para comenzar la fase de desarrollo de software.

4) Desarrollo de software

Los encargados de desarrollar software pueden instalar (o modificar y después instalar) software comprando a terceros o escribir programas diseñados a la medida del solicitante. La elección depende del costo de cada alternativa, del tiempo disponible para escribir el software y de la disponibilidad de los programadores. Por regla general, los programadores (o analistas - programadores) que trabajan en las grandes organizaciones pertenecen a un grupo permanente de profesionistas. En empresas pequeñas, donde no hay programadores, se pueden contratar servicios externos al efecto.

Los programadores también son responsables de la documentación y de proporcionar una explicación de cómo y por qué ciertos procedimientos se codifican en determinada forma. La documentación es esencial para probar el programa y llevar a cabo el mantenimiento una vez que la aplicación se encuentra instalada.

5) Pruebas al sistema

Durante la fase de pruebas del sistema, éste se emplea de manera experimental para asegurarse de que el software no tenga fallas, es decir, que funciona de acuerdo con las especificaciones y en la forma en que los usuarios esperan que lo haga; se alimenta con un conjunto de datos de prueba para su procesamiento y después se examinan los resultados, en ocasiones se permite que varios usuarios utilicen el sistema para que los analistas observen si tratan de emplearlo en formas no previstas. Siempre es preferible descubrir cualquier sorpresa antes de que la organización implante el sistema y dependa de él.

6) Implementación y evaluación

La implementación es el proceso de verificar y adecuar nuevo equipo, capacitar a los usuarios, instalar la aplicación y construir todos los archivos de datos necesarios para utilizarla. Una vez instaladas, las aplicaciones se emplean durante muchos años. Sin embargo, las organizaciones y los usuarios cambian con el paso del tiempo, por consiguiente, es indudable que debe dárseles

mantenimiento, realizando modificaciones al software, archivos o procedimientos para satisfacer las nuevas necesidades de los usuarios. Dado que los sistemas experimentan cambios de manera continua, los sistemas de información deben mantenerse siempre al día; en este sentido la implementación es un proceso en constante evolución.

La evaluación de un sistema se lleva a cabo para identificar puntos débiles y fuertes, desafortunadamente, no siempre recibe la atención que merece. Sin embargo, si se conduce en forma adecuada proporciona mucha información que puede ayudar a mejorar la efectividad de los esfuerzos de desarrollo de aplicaciones subsecuentes.

Método por análisis estructurado

Muchos especialistas en sistemas de información reconocen la dificultad de comprender, cabalmente, sistemas grandes y complejos. El método del desarrollo del análisis estructurado incorpora elementos tanto de análisis como de diseño. Además tiene como finalidad superar esta dificultad por medio de:

a) La división del sistema en componentes
b) La construcción de un modelo del sistema

Cuando los analistas comienzan a trabajar en un proyecto, a menudo tiene que profundizar en un área de la organización con la que tienen poca familiaridad. A pesar de esto, deben desarrollar un sistema que ayude a los futuros usuarios de un área específica. Cualquier nuevo sistema o conjunto de recomendaciones para cambios en el ya existente, sea manual o automatizado, debe conducir hacia una mejora. Para alcanzar este resultado, se espera que los analistas hagan lo siguiente:

- Conozcan los detalles y procedimientos del sistema en uso
- Obtengan una idea de las demandas futuras de la organización como resultado de crecimiento, aumento de la competencia en el mercado, cambios en las necesidades de consumidores, evolución de las estructuras financieras, introducción de nueva tecnología y cambios en las políticas, entre otros.
- Documentar los detalles del sistema actual para su revisión y discusión por otros.

    - Evaluar la eficiencia y efectividad del sistema actual y sus procedimientos, tomando en cuenta el impacto sobre las demandas anticipadas para el futuro.
- Recomendar todas las revisiones y ampliaciones del sistema actual, señalando su justificación. Si es apropiado, quizá se realice la propuesta de uno mayor e integral.
- Documentar las características con un nivel de detalle que permita comprender a otros sus componentes y de manera que permita manejar el desarrollo de este nuevo sistema.
- Fomentar la participación de gerentes y empleados en todo el proceso, tanto para aprovechar su experiencia y conocimiento del sistema actual, como para conocer sus ideas y opiniones relacionadas con los requerimientos del nuevo o de los cambios para el actual

1. Análisis estructurado

El análisis estructurado se concentra en especificar lo que se requiere que haga el sistema o la aplicación. No se establece cómo se cumplirán los requerimiento o la forma en que se implantara la aplicación. Más bien permite que las personas observen los elementos lógicos (lo que hará el sistema) separados de los componentes físicos (computadoras, terminales, sistemas de almacenamiento, etc.). Después de esto de puede desarrollar un diseño físico para la situación donde será utilizado.

Es un método para el análisis de sistemas manuales o automatizados, que conduce al desarrollo de especificaciones para los que son nuevos o para efectuar modificaciones a los ya existentes. Cuando los analistas de sistema abordan una situación poco familiar, siempre existe una pregunta sobre dónde comenzar el análisis. Una situación dinámica siempre puede ser vista como abrumadora debido a que muchas de las actividades se llevan a cabo constantemente. El análisis estructurado permite al analista conocer un sistema o proceso (actividad) en una forma lógica y manejable, al mismo tiempo que proporciona la base para asegurar que no se omite ningún detalle pertinente.

Sus elementos esenciales son símbolos gráficos, diagramas de flujo de datos y el direccionamiento centralizado de estos. En lugar de palabras, el análisis estructurado utiliza símbolos o iconos, para crear un modelo gráfico del sistema. Los modelos de este tipo muestran los detalles del sistema pero sin introducir procesos manuales o computarizados, archivos o procedimientos. Si se seleccionan símbolos y notaciones correctas entonces casi cualquier persona puede seguir la forma en que los componentes se acomodarán entre sí para formar el sistema.

Los iconos identifican los elementos básicos de los procesos, el flujo de datos, y sitio donde se almacenan, y fuentes y destinos de éstos. Se dibuja una línea alrededor del sistema para señalar qué elementos se encuentran dentro de el y cual es su frontera. El diagrama lógico de flujo de datos muestra las fuentes y destinos de los mismos, identifica y da nombre a los procesos que se llevan a cabo, como se muestra en la figura 1, así como a los grupos de datos que relacionan una función con otra y señala los almacenes a los que tiene acceso.

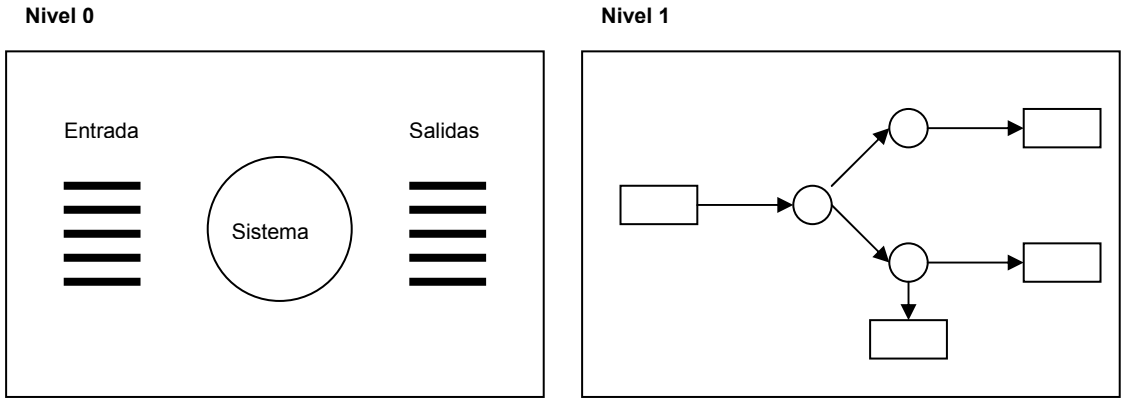


Figura 1: Diagrama de flujo de datos

El modelo del sistema recibe el nombre de diagrama de flujo de datos (DFD). La descripción completa de un sistema está formada por un conjunto de diagramas de flujo de datos.

Para desarrollar una descripción de un sistema por el método del análisis estructurado se sigue un proceso descendente (TOP down). El modelo original se detalla en un diagrama de bajo nivel que muestra características adicionales del sistema; cada proceso puede desglosarse en diagramas de flujo de datos cada vez más detallados, esta secuencia se repite hasta que se obtienen suficientes detalles que permiten al analista comprender en su totalidad la parte del sistema que se encuentra bajo investigación. Nótese que el enfoque está sobre los datos y procesos. No se hace mención alguna de computadoras, comunicaciones, personas o departamentos y tampoco se incluyen detalles físicos.

Todas las definiciones de los elementos en el sistema (flujo de datos, procesos y almacenes) están descritos en forma detallada en un diccionario de datos. Si algún miembro del equipo encargado del proyecto desea saber alguna definición del nombre de un dato o el contenido particular de un flujo, esta información debe encontrarse disponible en el diccionario de datos.

2. Diseño estructurado

El diseño estructurado, otro elemento del análisis estructurado que emplea la descripción gráfica, se enfoca al desarrollo de especificaciones de software. La meta del diseño estructurado es crear programas formados por módulos independientes unos de otros desde el punto de vista funcional. Este enfoque no sólo conduce hacia mejores programas sino que facilita el mantenimiento de los mismos cuando surja la necesidad de hacerlo.

El diseño estructurado es una técnica específica para el diseño de programas y no un método de diseño de comprensión. Es decir, no indica nada relacionado con el diseño de archivos o base de datos, la presentación de entradas o salidas, la secuencia de procesamiento o el hardware que dará soporte a la aplicación. Esta técnica conduce a la especificación de módulos de programa que son funcionalmente independientes.

Su herramienta fundamental el diagrama estructurado. Al igual que los diagramas de flujo de datos, los diagramas estructurados son de naturaleza gráfica y evitan cualquier referencia relacionada con el hardware o detalles físicos. Su finalidad no es mostrar la lógica de los programas (que es la tarea de los diagramas de flujo). Los diagramas estructurados describen la interacción entre módulos independientes junto con los datos que de un módulo pasan a otro (figura 2). Estas especificaciones funcionales para los módulos se proporcionan a los programadores antes de que dé comienzo la fase de escritura de código.

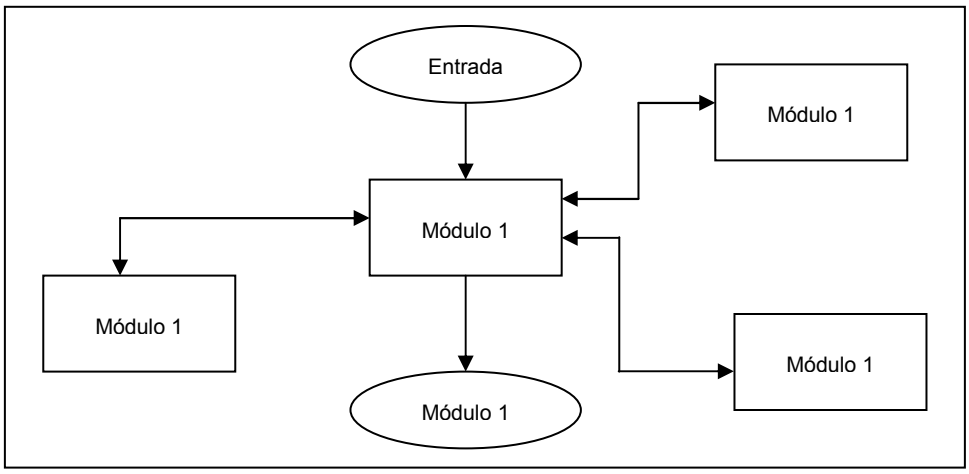


Figura 2: Diagrama estructurado

3. Empleo del análisis estructurado con otros métodos

Se combina, con bastante frecuencia, con el método del ciclo de vida clásico de desarrollo de sistemas. Por ejemplo los analistas pueden optar por desarrollar diagramas de flujo de datos como una forma para documentar las relaciones entre componentes, durante la investigación detallada de algún sistema existente. Asimismo, se pueden definir los archivos y datos de un diccionario centralizado de datos de acuerdo con las reglas del análisis estructurado.

Los analistas deciden con frecuencia que el desarrollo de diagramas y esquemas es una tarea que consume mucho tiempo, sobre todo si el sistema es grande y complejo. (Común que los diagramas tengan que dibujarse una y otra vez conforme se adquiera nueva información)

Método del prototipo de sistemas

Este método hace que el usuario participe de manera más directa en la experiencia de análisis y diseño que cualquiera de los métodos ya presentados. La construcción de prototipos es muy eficaz bajo las circunstancias correctas. Sin embargo, al igual que los otros métodos, es útil si se emplea en el momento adecuado y en la forma apropiada.

El prototipo: Es un sistema que se ha desarrollado con la finalidad de probar ideas y suposiciones relacionadas con el nuevo sistema. Está constituido por software que acepta entradas, realiza cálculos, produce información o que lleva a cabo otras actividades significativas. Es la primera versión o iteración de un sistema de información en el modelo original. (Figura 3)

Los usuarios evalúan el diseño y la información generada por el sistema. Lo anterior sólo puede hacerse con efectividad si los datos utilizados, al igual que las situaciones, son reales. Por otra parte, deben esperarse cambios a medida que el sistema es utilizado.

Etapas del método de prototipos: El desarrollo de un prototipo para una aplicación se lleva a cabo en una forma ordenada, sin importar las herramientas utilizadas.

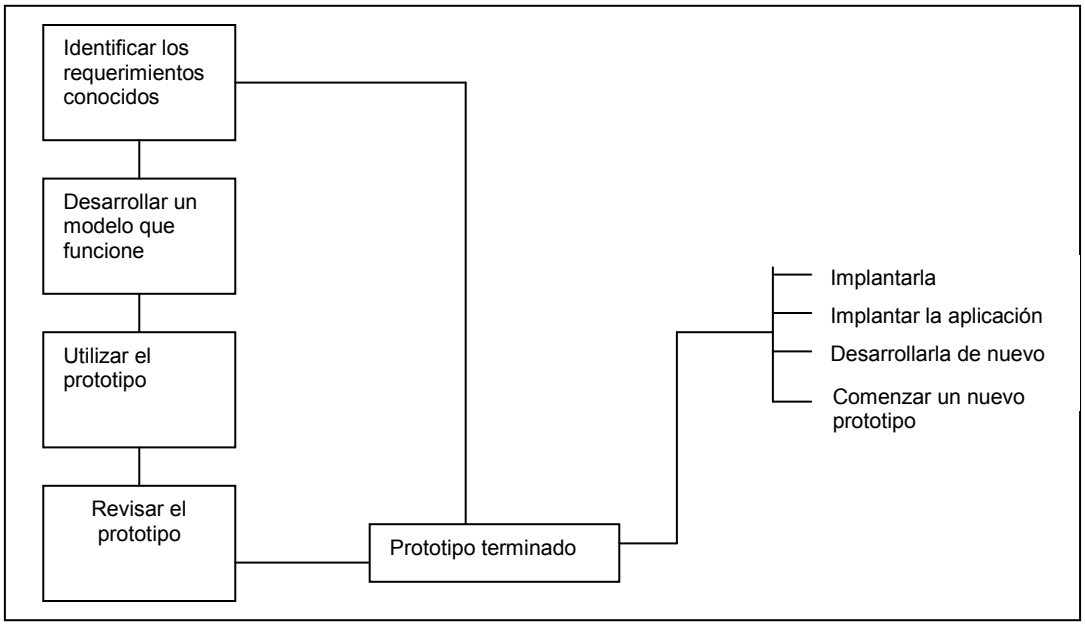


Figura 3: Esquema general del desarrollo de prototipos

a) Identificación de requerimientos conocidos

La determinación de los requerimientos de una aplicación es tan importante para el método de desarrollo de prototipos como lo es para los métodos del ciclo clásico de vida de desarrollo de sistemas o análisis estructurado (aunque las técnicas son diferentes). Por consiguiente, antes de crear un prototipo, los analistas y usuarios deben trabajar juntos para identificar los requerimientos conocidos que tienen que satisfacerse; para hacerlo determinan los fines para los que servirá el sistema y el alcance de sus capacidades. En general, sólo un analista de sistemas es el que coordina este paso.

b) Desarrollo de un modelo de trabajo que funcione

La construcción de un prototipo es un proceso iterativo de desarrollo. Antes de la primera iteración, los analistas de sistemas explican el método a los usuarios, las actividades a realizar, la secuencia en que se llevarán acabo y también discuten las responsabilidades de cada participante. Es útil comenzar el proceso de construcción del prototipo con el desarrollo de un plan general que permita a las personas conocer lo que se espera de ellas y del proceso de desarrollo. Un cronograma para el inicio y fin de la primera iteración es de gran ayuda y por lo tanto debe elaborarse justo antes de comenzar las actividades. Sin embargo, dada la naturaleza de este método de desarrollo, es difícil y en ocasiones imposible, fijar una fecha tentativa de terminación. La experiencia con el sistema es la que determina eventualmente cuando el sistema está acabado. Para comenzar la primera iteración,

Usuarios y analistas identifican de manera conjunta los datos que son necesarios para el sistema y especifican la salida que debe producir la aplicación. Las decisiones de diseño necesarias para desarrollar la salida del sistema cambian muy poco en relación con las tomadas en otros métodos de desarrollo. Sin embargo, con un prototipo, se espera que las especificaciones iniciales estén incompletas. En general, se necesitan entre dos y tres reuniones para establecer aquellas.

Asimismo, el analista estima los costos asociados con el desarrollo del prototipo. Este paso es muy importante, aunque sólo se indique una estimación. Lo anterior da a la administración y a los participantes una idea de los gastos necesarios (personal, equipo y artículos de consumo) que les permitan revisar el plan de desarrollo.

En el desarrollo de un prototipo se preparan los siguientes componentes:

a) El lenguaje para el diálogo o conversación entre el usuario y el sistema
b) Pantallas y formatos para entrada de datos
c) Módulos esenciales de procesamiento
d) Salida del sistema

Al construir el prototipo se deben seguir los estándares para datos que emplea la organización. La incorporación en la interfaz entrada/salida de características representativas, que serán incluidas en el sistema final permite una mayor exactitud en el proceso de evaluación.

En esta fase no se prepara la documentación ni tampoco las especificaciones de salida o de diseño de software, debido a que es más importante la rapidez con la que se construye el prototipo que la eficiencia de operación.

c) Utilización del prototipo

Es responsabilidad del usuario trabajar con el prototipo y evaluar sus características y operación. La experiencia con el sistema bajo condiciones reales permite obtener la familiaridad indispensable para determinar los cambios o mejoras que sean necesarios, así como la eliminación de características inadecuadas o innecesarias.

d) Revisión del prototipo

Durante la evaluación, los analistas de sistemas desean capturar información sobre lo que les gusta y lo que les desagrada a los usuarios; al mismo tiempo anotan por qué reaccionan en la forma en que lo hacen. La información obtenida tendrá influencia sobre las características de la siguiente versión de la aplicación. Asimismo, la evaluación permite profundizar en los rasgos de los usuarios y también en los de la empresa.

Los cambios al prototipo son planificados con los usuarios antes de llevarlos a cabo. Sin embargo, el analista es el responsable de realizar las modificaciones.

e) Repetición del proceso las veces que sea necesario

El proceso antes descrito se repite varias veces, finaliza cuando los usuarios y analistas están de acuerdo en que el sistema ha evolucionado lo suficiente como para incluir todas las características necesarias o cuando ya es evidente que no se obtendrá mayor beneficio con una iteración adicional.

f) Uso de prototipos.

Cuando el prototipo está terminado el siguiente paso es tomar la decisión sobre cómo proceder. Existen cuatro caminos a seguir después de evaluar la información obtenida con el desarrollo y uso del prototipo:

- Abandono de la aplicación.

  En algunos casos, la decisión es descartar el prototipo y abandonar el desarrollo de la aplicación. Esta conclusión no significa que fuese un error emprender el proceso de desarrollo del prototipo o un desperdicio de recursos. Más bien, la información y experiencia ganada con su empleo del prototipo condujo hacia una decisión de desarrollo. Es probable que usuarios y analistas hayan aprendido que el sistema era innecesario o hayan descubierto otra solución durante el proceso, o quizá el enfoque no fue el más adecuado

- Implementación del prototipo.

  La segunda opción es implantar el prototipo. Algunas veces este se convierte en el sistema que se necesita. En este caso, se implanta sin ninguna modificación y no se emprenden más esfuerzos de desarrollo. Esta decisión es más probable de tomarse bajo una o más de las siguientes circunstancias:

- La evolución del prototipo condujo a una aplicación que tiene las características, capacidades y desempeños requeridos.
- La aplicación será utilizada con poca frecuencia y no es importante su rapidez o eficiencia operacional
- La aplicación no tiene efectos sobre otras aplicaciones o datos de la organización y tampoco interaccionan con ellos; además satisface las necesidades de los usuarios inmediatos
- Su medio ambiente se encuentra en un estado de flujo; es difícil determinar necesidades a largo plazo o condiciones de operación más estables. En consecuencia no es posible justificar otras actividades de desarrollo, el prototipo es de utilidad para las condiciones actuales

- Re - desarrollo de la aplicación

Cuando un prototipo tiene éxito puede proporcionar información muy amplia con respecto a los requerimientos de la aplicación y conducir a su completo desarrollo. Terminar el prototipo no significa finalizar el proceso de desarrollo. Más bien señala el comienzo de la siguiente actividad: el desarrollo completo de la aplicación.

La información recopilada sugiere características que deben añadirse a la aplicación. También permite que los analistas y usuarios tengan mayor información para tomar decisiones relacionadas con la forma en que debe realizarse el procesamiento y la información que se debe producir. La presentación de información, incluida la forma de interactuar con el sistema, es un requerimiento importante; los datos obtenidos a partir de la experiencia con el prototipo contribuyen a considerar con mayor eficacia otras características adicionales que son necesarias.

Con frecuencia, cuando la aplicación se vuelve a desarrollar, se pone mucha atención en el mejor uso posible de los recursos del sistema. La velocidad de procesamiento y el tiempo de respuesta tienen mayor importancia, al igual que el uso eficiente de los medios de almacenamiento.

Las dos formas más comunes de incorporar la construcción de un prototipo para la aplicación son las siguientes:

- El prototipo se emplea como una opción para la determinación de requerimientos; sus características del prototipo son consideradas tanto como los requerimientos a satisfacer en las subsecuentes actividades de desarrollo
- El prototipo se utiliza como sustituto para el diseño e implementación de la aplicación, es decir, como un esqueleto a partir del que se construye el resto del sistema
- Inicio de un nuevo prototipo

Algunas veces la información ganada con el desarrollo y uso del prototipo, sugiere el empleo de un enfoque muy diferente para satisfacer las necesidades de la organización. En este caso, es posible encontrar que las características de la aplicación son muy diferentes si el prototipo es inadecuado para demostrarlas y evaluarlas.

Como consecuencia, más que emprender directamente el esfuerzo de desarrollo total, la gerencia puede apoyar la creación de otro prototipo que permita añadir más información a la que ya se tiene disponible.

De nuevo, es importante notar que no se han desperdiciado los esfuerzos hasta este momento, aunque ellos revelen requerimientos diferentes a los anticipados. Conocerlos en esta fase del proceso, justo cuando se cambia la dirección del esfuerzo de desarrollo del prototipo, es mucho mejor que advertirlos después de invertir en él todo el esfuerzo de desarrollo e implementación.

Metodología UML para programación orientada a objetos

Actualmente la tecnología orientada a objetos ya no se aplica solamente a los lenguajes de programación, además se está aplicando en el análisis y diseño con mucho éxito, al igual que en las bases de datos. Una buena programación orientada a objetos no sólo queda en desarrollar todo el sistema aplicando esta tecnología, de ahí la importancia del análisis y el diseño orientado a objetos.

La programación orientada a objetos ahora es una de las formas más populares de programar y está siendo muy aceptado en el desarrollo de proyectos de software desde los últimos años. Esta aceptación se debe a sus grandes capacidades y ventajas frente a las antiguas formas de programar.

Antecedentes de la programación de la programación orientada a objetos

Tradicionalmente, la programación fue hecha en una manera secuencial o lineal, es decir una serie de pasos consecutivos con estructuras consecutivas y bifurcaciones.

Los lenguajes basados en esta forma de programación ofrecían ventajas al principio, pero el problema ocurre cuando los sistemas se vuelven complejos. Estos programas escritos al estilo "espaguetti" no ofrecen flexibilidad y el mantener una gran cantidad de líneas de código en sólo bloque se vuelve una tarea complicada, mucho más para la integración de nuevas funciones o correcciones en el código de programación.

Frente a esta dificultad aparecieron los lenguajes basados en la programación estructurada. La idea principal de esta forma de programación es separar las partes complejas del programa en módulos o segmentos que sean ejecutados conforme se requieran. De esta manera tenemos un diseño modular, compuesto por módulos independientes que puedan comunicarse entre sí. Poco a poco este estilo de programación fue reemplazando al estilo "espaguetti" impuesto por la programación lineal.
La evolución que se fue dando en la programación estructurada se orientaba siempre a ir descomponiendo más el programa. Este tipo de descomposición conduce directamente a la programación orientada a objetos.

La tendencia de crear programas cada vez más grandes y complejos llevó a los desarrolladores a crear una nueva forma de programar que les permita crear sistemas de niveles empresariales y con reglas de negocios muy complejas. Para estas necesidades ya no bastaba la programación estructurada ni mucho menos la programación lineal. Es así como aparece la programación orientada a objetos (POO). La POO viene de la evolución de la programación estructurada; básicamente la POO simplifica la programación con la nueva filosofía y nuevos conceptos que tiene. La POO se basa en dividir el programa en pequeñas unidades lógicas de código. A estas pequeñas unidades lógicas de código se les llama objetos. Los objetos son unidades independientes que se comunican entre ellos mediante mensajes.

Conceptos básicos de la programación orientada a objetos

Objeto

Primero comencemos entendiendo que es un objeto del mundo real. Un objeto del mundo real es cualquier cosa que vemos a nuestro alrededor. Por ejemplo, un automóvil, un árbol, una computadora, etc.

Analicemos un poco más a un objeto del mundo real, como la computadora. No necesitamos ser expertos en hardware para saber que una computadora está compuesta internamente por varios componentes: la tarjeta madre, el chip del procesador, un disco duro, una tarjeta de video, y otras partes más. El trabajo en conjunto de todos estos componentes hace operar a una computadora.

Cada componente es una unidad autónoma, y todo lo que necesitamos saber de adentro es cómo interactúan entre sí los componentes, saber por ejemplo si el procesador y las memorias son compatibles con la tarjeta madre, o conocer donde se coloca la tarjeta de video. Cuando conocemos como interaccionan los componentes entre sí, podremos armar fácilmente una computadora.

La programación orientada a objetos trabaja de esta manera. Todo el programa está construido en base a diferentes componentes (Objetos), cada uno tiene un rol específico en el programa y todos los componentes pueden comunicarse entre ellos de formas predefinidas.

Todo objeto del mundo real tiene 2 componentes: características y comportamiento.
Por lo tanto podemos definir que un objeto es una unidad de código compuesto de variables y métodos relacionados.

Clase

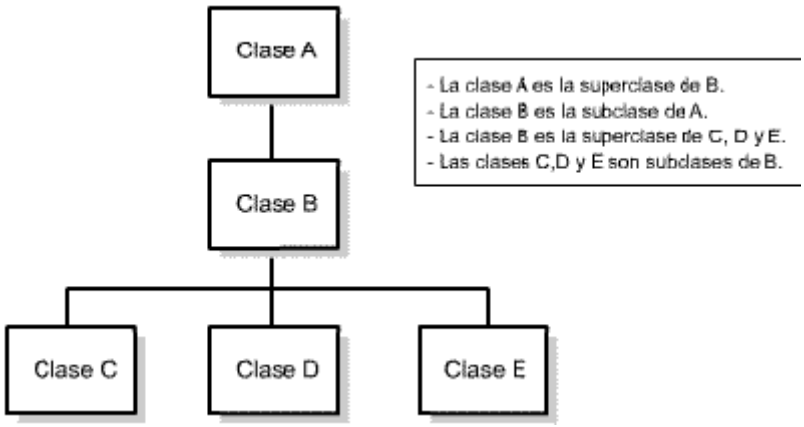
En el mundo real, normalmente tenemos muchos objetos del mismo tipo. Por ejemplo, nuestro teléfono celular es sólo uno de los miles que hay en el mundo. Si hablamos en términos de la programación orientada a objetos, podemos decir que nuestro objeto celular es una instancia de una clase conocida como "celular". Los celulares tienen características (marca, modelo, sistema operativo, pantalla, teclado, etc.) y comportamientos (hacer y recibir llamadas, enviar mensajes multimedia, transmisión de datos, etc.).

Cuando se fabrican los celulares, los fabricantes aprovechan el hecho de que los celulares comparten esas características comunes y construyen modelos o plantillas comunes, para que a partir de esas se puedan crear muchos equipos celulares del mismo modelo. A ese modelo o plantilla le llamamos CLASE, y a los equipos que sacamos a partir de ella la llamamos OBJETOS.
La clase es un modelo o prototipo que define las variables y métodos comunes a todos los objetos de cierta clase. También se puede decir que una clase es una plantilla genérica para un conjunto de objetos de similares características.

Herencia

La herencia es uno de los conceptos más cruciales en la POO. La herencia básicamente consiste en que una clase puede heredar sus variables y métodos a varias subclases (la clase que hereda es llamada superclase o clase padre). Esto significa que una subclase, aparte de los atributos y métodos propios, tiene incorporados los atributos y métodos heredados de la superclase. De esta manera se crea una jerarquía de herencia.

En general, podemos tener una gran jerarquía de Clases tal y como vemos en el siguiente gráfico:



Mensaje

Un objeto es inútil si está aislado. El medio empleado para que un objeto interactúe con otro son los mensajes. Hablando en términos un poco más técnicos, los mensajes son invocaciones a los métodos de los objetos.

Características asociadas a la programación orientada a objetos POO

Abstracción

La abstracción consiste en captar las características esenciales de un objeto, así como su comportamiento. Pensemos en el objeto automóvil ¿Qué características podemos abstraer de los automóviles? O lo que es lo mismo ¿Qué características semejantes tienen todos los automóviles? Todos tendrán una marca, un modelo, número de chasis, peso, llantas, puertas, ventanas, etc. Y en cuanto a su comportamiento todos los automóviles podrán acelerar, frenar, retroceder, etc.

Encapsulamiento

El encapsulamiento consiste en unir en la Clase las características y comportamientos, esto es, las variables y métodos. Es tener todo esto es una sola entidad. En los lenguajes estructurados esto era imposible. Es evidente que el encapsulamiento se logra gracias a la abstracción y el ocultamiento que veremos a continuación.

La utilidad del encapsulamiento va por la facilidad para manejar la complejidad, ya que tendremos a las Clases como cajas negras donde sólo se conoce el comportamiento pero no los detalles internos, y esto es conveniente porque nos interesará será conocer qué hace la Clase pero no será necesario saber cómo lo hace.

Ocultamiento

Es la capacidad de ocultar los detalles internos del comportamiento de una Clase y exponer sólo los detalles que sean necesarios para el resto del sistema. El ocultamiento permite 2 cosas: restringir y controlar el uso de la Clase. Restringir porque habrá cierto comportamiento privado de la Clase que no podrá ser accedido por otras Clases. Y controlar porque daremos ciertos mecanismos para modificar el estado de nuestra Clase y es en estos mecanismos dónde se validarán que algunas condiciones se cumplan. En Java el ocultamiento se logra usando las palabras reservadas: public, private y protected delante de las variables y métodos.

Ventajas de la Programación Orientada a Objetos POO

- Fomenta la reutilización y extensión del código.
- Permite crear sistemas más complejos.
- Relacionar el sistema al mundo real.
- Facilita la creación de programas visuales.

    - Construcción de prototipos
    - Agiliza el desarrollo de software
- Facilita el trabajo en equipo
- Facilita el mantenimiento del software

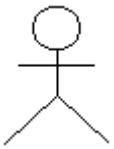
Definición de UML

UML (Unified Modeling Language) es un lenguaje que permite modelar, construir y documentar los elementos que forman un sistema software orientado a objetos. Se ha convertido en el estándar de facto de la industria, debido a que ha sido impulsado por los autores de los tres métodos más usados de orientación a objetos: Grady Booch, Ivar Jacobson y Jim Rumbaugh. Estos autores fueron contratados por la empresa Rational Software Co. para crear una notación unificada en la que basar la construcción de sus herramientas CASE. En el proceso de creación de UML han participado, no obstante, otras empresas de gran peso en la industria como Microsoft, Hewlett-Packard, Oracle o IBM, así como grupos de analistas y desarrolladores.

Esta notación ha sido ampliamente aceptada debido al prestigio de sus creadores y debido a que incorpora las principales ventajas de cada uno de los métodos particulares en los que se basa (principalmente Booch, OMT y OOSE). UML ha puesto fin a las llamadas "guerras de métodos" que se han mantenido a lo largo de los 90, en las que los principales métodos sacaban nuevas versiones que incorporaban las técnicas de los demás. Con UML se fusiona la notación de estas técnicas para formar una herramienta compartida entre todos los ingenieros software que trabajan en el desarrollo orientado a objetos. Uno de los objetivos principales de la creación de UML era posibilitar el intercambio de modelos entre las distintas herramientas CASE orientadas a objetos del mercado. Para ello era necesario definir una notación y semántica común.

Diagrama de Casos de Uso

Un diagrama de Casos de Uso muestra las distintas operaciones que se esperan de una aplicación o sistema y cómo se relaciona con su entorno (usuarios u otras aplicaciones). Se representa en el diagrama por una elipse, denota un requerimiento solucionado por el sistema. Cada caso de uso es una operación completa desarrollada por los actores y por el sistema en un diálogo. El conjunto de casos de uso representa la totalidad de operaciones desarrolladas por el sistema. Va acompañado de un nombre significativo. En el caso del ejemplo se tienen como casos de uso de la cafetera RecibirDinero, PedirAzucar, PedirProducto, DarVueltas y Cancelar.

Actor Es un usuario del sistema, que necesita o usa algunos de los casos de uso. Se representa mediante un  , acompañado de un nombre significativo, si es necesario.

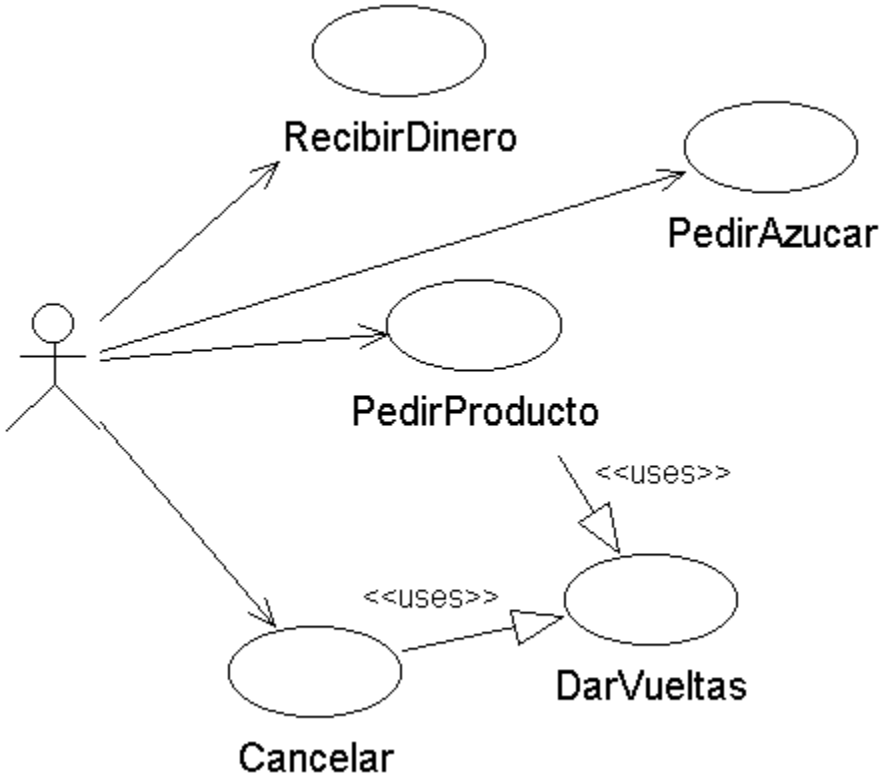
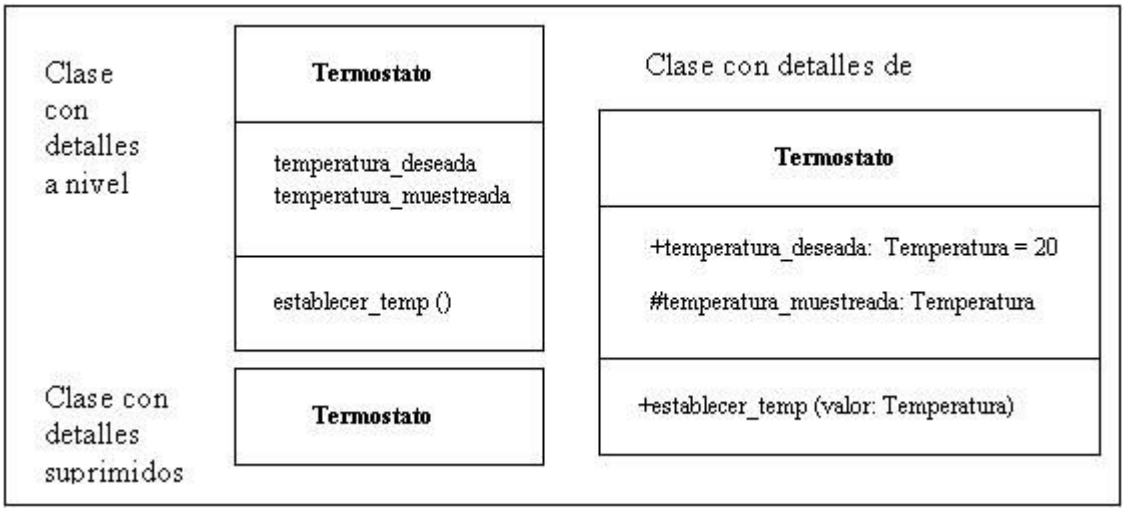


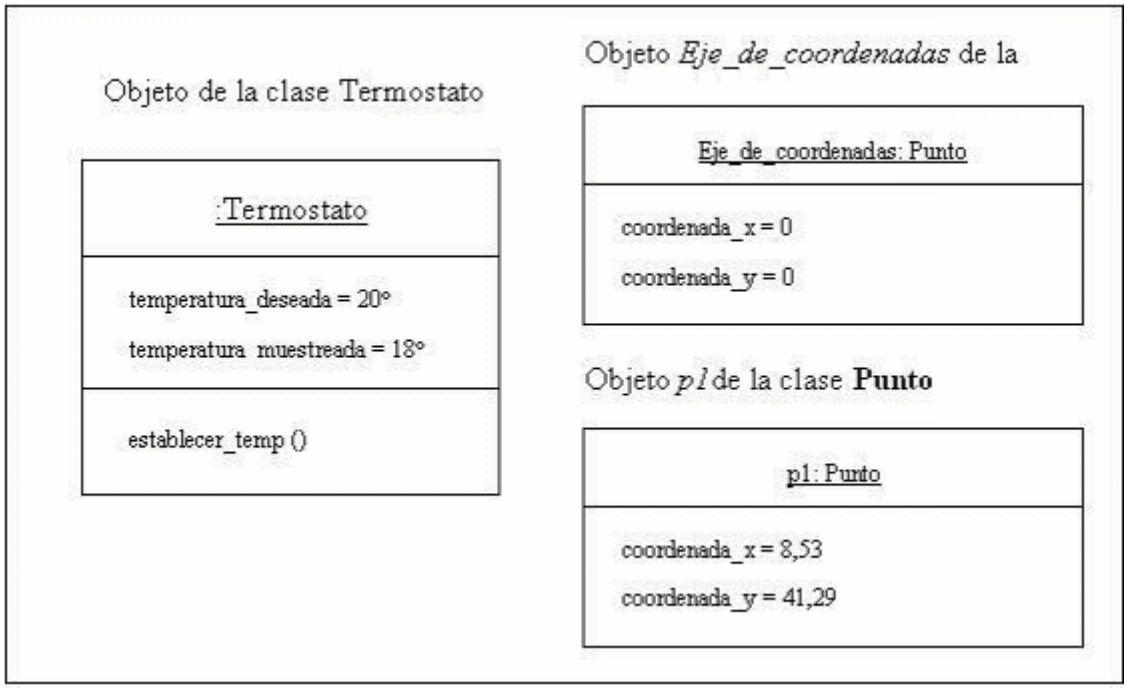
Diagrama de Estructura Estática

Los Diagramas de Estructura Estática de UML se van a utilizar para representar tanto Modelos Conceptuales como Diagramas de Clases de Diseño. Ambos usos son distintos conceptualmente, mientras los primeros modelan elementos del dominio los segundos presentan los elementos de la solución software. Ambos tipos de diagramas comparten una parte de la notación para los elementos que los forman (clases y objetos) y las relaciones que existen entre los mismos (asociaciones). Sin embargo, hay otros elementos de notación que serán exclusivos de uno u otro tipo de diagrama.

Clase Una clase se representa mediante una caja subdividida en tres partes: En la superior se muestra el nombre de la clase, en la media los atributos y en la inferior las operaciones. Una clase puede representarse de forma esquemática, con los atributos y operaciones suprimidos, siendo entonces tan solo un rectángulo con el nombre de la clase.



Objeto Un objeto se representa de la misma forma que una clase. En el compartimento superior aparecen el nombre del objeto junto con el nombre de la clase subrayados, según la siguiente sintaxis: nombre_del_objeto: nombre_de_la_clase Puede representarse un objeto sin un nombre específico, entonces sólo aparece el nombre de la clase.



Asociaciones

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

Bases de Datos

Aspectos de las bases de datos

Las bases de datos son componentes esenciales de los sistemas de información, usadas en forma rutinaria desde las computadoras menores hasta las supercomputadores. El diseño de las bases de datos utilizada no solo por los profesionales sino también por los no especialistas que buscan los mejores mecanismos para resguardar su información.

Evolución de las Bases de Datos

La expresión base de datos comenzó a popularizarse al principio del decenio de los 60's, antes de esta época se hablaba de archivos y de conjunto de datos. Para esas fechas, surge COBOL y los primeros DBMS que se encontraban basados en metodología de jerarquía y de red.

Aproximadamente en 1965, las bases de datos penetraban por primera vez como una herramienta al respaldo de datos y antes de que aparecieran las computadoras de la tercera generación (1965) el software ejecutaba las operaciones de entrada/salida de los dispositivos de almacenamiento. La codificación incluida en los programas de aplicación se encargaba de la organización de los datos, y esto de manera muy elemental, por lo general solo servía a modo de simples archivos secuenciales en cinta. Entre lo que podemos destacar de IBM está el sistema conocido como SABRE que tenía una vista de dos niveles, conceptual y del usuario, la organización de los datos que constituyen así el fundamento para el modelo jerárquico. No había independencia de datos y para actualizar un archivo, había que escribir de nuevo. La mayoría de los archivos servían solo para una aplicación. Y esto involucraba mucho trabajo ya que sé tenia que volver a generar el archivo y no existía el mecanismo de actualizarlo, por tanto era tedioso trabajar con este tipo de bases de datos.

En esta primera etapa se utilizaban también algunos dispositivos de acceso al azar, lo que permitía al usuario el entrar a cualquiera de los registros almacenados sin tener que explorar forzosamente todo el archivo.

En la segunda etapa se reconoció la naturaleza cambiante de los archivos y de los dispositivos de almacenamiento. El efecto de los cambios que se introducían en el hardware y el software hizo posible modificar la distribución física de los datos sin que por ello se alterase su estructura lógica,

siempre que no se introdujesen cambios en los contenidos de los registros ni en la estructura fundamental de los archivos.

Los archivos utilizados durante esta segunda etapa, estaban por lo general diseñados, como los de la primera, de modo que podían servir con la mayor eficiencia posible las necesidades de solo una aplicación, así que para cualquiera otra, no había más remedio que crear nuevos archivos, con pasadas de computadora independientes.

Por la década de los 70 surge el modelo relacional es decir la definición, de lo que conocemos como matemática del álgebra relacional.

Después de la mitad de la década surge ORACLE se establece y realiza la primera implementación real del modelo relacional para que a finales de esta década la misma compañía establezca el primer RBDMS comercial de ORACLE siendo la versión 2.0.

Ya en la década de los 80 y después de que se ha realizado cambios importantes se funda INFORMIX dando paso a la tercera generación de DBMS, siendo tan importante que se utiliza aun en la actualidad, ya en poco tiempo se establece SQL que es una de las principales herramientas de trabajo que actualmente se utilizan, hoy destacándose paralelamente el uso del modelo relacional, iniciándose así los esfuerzos de ANSI SQL y liberación de los primeros DBMS que podía ser operadas en Mainframe y computadora tipo PC utilizando el producto llamado DATABASE2 conocido en el mercado como DB2.

Antes de que termine la década se establece Dbase III y IV, se establecen las estructuras de consultas con un lenguaje de programación y con ello surgiendo las herramientas de SQL de Oracle en versión 6.0 y se empieza a utilizar como herramienta de base de datos a SQL Server por parte de las compañías Microsoft y Sybase.

Es importante destacar que en los últimos años de la década de los 80 y hasta la mitad de los 90 crece en forma importante el análisis, diseño y desarrollo de bases de datos utilizando las herramientas de SQL que permiten manipular la información en una forma natural. El nacimiento del diseño de sistemas orientado a objetos y con las herramientas que apoyan a las comunicaciones con las bases de datos como es ODBC permitió poder manejar diferentes bases de datos y utilizar solo un mediador.

Ya en la década de los 90 las herramientas alcanzan niveles importantes de almacenamiento y ello acompañado por la tecnología de computadoras que permite tener una mayor operabilidad en tiempo y distribución de bases de datos. Así entonces en 1995 se cuenta con ODBC en otras

plataformas entre las que se destaca a ODBC para UNIX utilizando equipos SUN y HP y con herramientas de SQL para trabajar y operar información en diferentes plataformas.

Hoy en día se utilizan el diseño y estructura de base de datos con multimedia en vía de Internet o Intranet y se envuelve a gran numero de redes por las cuales a diario se distribuye información por sistemas operativos UNIX y herramientas de bases de datos como son Oracle versión 8. Desafortunadamente, las metodologias de diseño de las bases de datos y en especial la relacional no ha sido muy popular y paralelo a esto que no se realiza un estudio de la información antes de llevar a cabo el diseño ya que solo se enfoca al almacenamiento de la información, por lo anterior en gran medida se tiene que los organismos y personas profesionales confían poco en la eficiencia de la base de datos, básicamente en el diseño y esto ha llevado a diversos fracasos de sistemas y siendo una principal causa los proyectos de distribución o administración de información.

Debido a la falta de un enfoque estructurado y sin una metodología adecuada, se tienen bases de datos ineficientes e inadecuadas con relación a la aplicación además de la documentación deficiente de la estructura de la base de datos limita al mantenimiento del sistema y lo hace vulnerable.

Definición de bases de datos

Puede definirse como una colección de datos interrelacionados almacenados en conjunto sin redundancias perjudiciales o innecesarias; su funcionalidad es la de servir a una aplicación o más, de la mejor manera posible, los datos se almacenan dé modo que resulten independientes de los programas o sistemas desarrollados ya que los usan, se emplean métodos bien determinados para incluir datos nuevos para modificar o extraer los datos almacenados. Dícese que un sistema comprende una colección de datos (Figura 4) cuando estos son totalmente independientes desde el punto de vista estructural.

La principal idea en la implementación de una base de datos es la de que los mismos datos puedan ser utilizados y aprovechados por tantas aplicaciones como sea posible. Por eso, la base se concibe a menudo como un repositorio donde se reúne la información necesaria para el ejercicio de las funciones propias de un organismo.

Una base de datos permitirá la lectura de información almacenada y la modificación de los que son necesarios para el control de las operaciones.

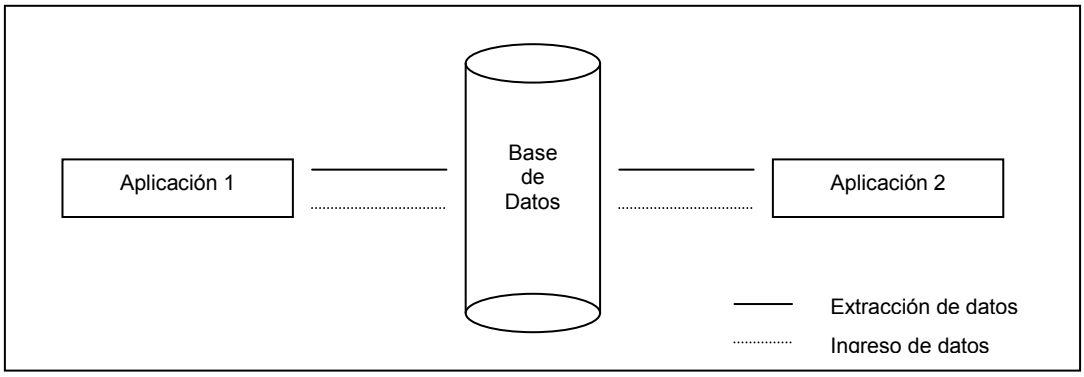


Figura 4: Definición de base de datos

Objetivos de base de datos

Al contar con un sistema de base de datos que logre proporcionar un control centralizado de información, los objetivos que se buscan son los siguientes:

a) ***Minimizar la redundancia***: Si bien las bases de datos pueden ayudar en forma importante al control de información y reducir el trabajo, en las bases mal diseñadas existen una gran cantidad de datos duplicados y esto hace que se eleve el costo de almacenamiento y de acceso, además de que se corre el riesgo de que exista inconsistencia es decir que exista mas de una copia y no concuerden entre sí.

Con las bases de datos se busca abatir la redundancia, aunque algunas veces se prefiere tener la redundancia controlada o mínima con el objeto de reducir tiempos de acceso o simplificar cuestiones de direccionamiento.

Al tener redundancia controlada se obtiene: Reducción de errores, datos consistentes, optimización del espacio de almacenamiento físico, y reducción de requerimientos de almacenamiento de respaldo.

b) ***Inconsistencia de los datos:*** Al eliminar o controlar la redundancia, la inconsistencia en los datos no ocurrirá, ya que cualquier cambio hecho en los datos duplicados actualizará automáticamente los mismos de la base de datos.

c) ***Compartir los datos****:* No solo las aplicaciones existentes pueden compartir información en la base de datos, sino que también es factible desarrollar nuevas aplicaciones que operen con los mismos datos. Es decir que la misma información sea utilizada por otras aplicaciones y se obtenga un máximo beneficio aunque los datos fuente pueden ser usados

concurrentemente por múltiples aplicaciones, todas las aplicaciones usaran los datos actualizados.

d) ***Cumplir normas establecidas***: Si se tiene un control centralizado de la base de datos, se puede garantizar que se cumplen todas las formas aplicables a la representación de los datos. Es deseable unificar los formatos de estos como ayuda para la migración e intercambio de información entre los sistemas.

e) ***Aplicar restricciones de seguridad***: No es recomendable que todos los usuarios de sistema de base de datos puedan tener acceso a toda la información, se deben definir controles de autorización es decir los usuarios podrán trabajar por niveles de acceso y en cada uno de estos se puede especificar el tipo de privilegio al que tienen derecho.

f) ***Conservar la integridad***: El objetivo de la integridad es garantizar que la información almacenada en la base de datos sea exacta es decir sea consistente, la integridad de los datos combinada con una redundancia controlada brinda la habilidad de reconstruirlos completamente los datos si existen fallas en el hardware o el software. Se deben incluir procedimientos de chequeo que aseguren que los valores que se van a depositar en la base de datos se ajusten a ciertas reglas establecidas.

g) ***Abstracción de la información***: Uno de los principales objetivos de un sistema de bases de datos es proporcionar a los usuarios una visión abstracta de la información, esto es, el sistema oculta ciertos detalles relativos a la forma en como los datos se almacenan y mantienen. Sin embargo, para que el sistema sea útil, la información debe recuperarse en forma eficiente.

Los niveles de abstracción (Figura 5) en los que puede observarse la base de dato[1] son:

- *Nivel físico o interno* (Modelo físico): Este es el nivel mas bajo de abstracción, el cual tiene por objetivo la representación y distribución física de los datos y la organización de estos en las unidades de almacenamiento.
- *Nivel conceptual* (Modelo conceptual): En este nivel se describen los datos reales que están almacenados en la base de datos y las relaciones existentes entre los mismos.
- *Nivel de visión o externo* (Modelo lógico): El nivel de visión o externo es el más cercano a los usuarios, es decir la manera en que cada usuario ve los datos.

[1] F. Korth Henry, Fundamentos de Bases de Datos, pág. 4 y 5

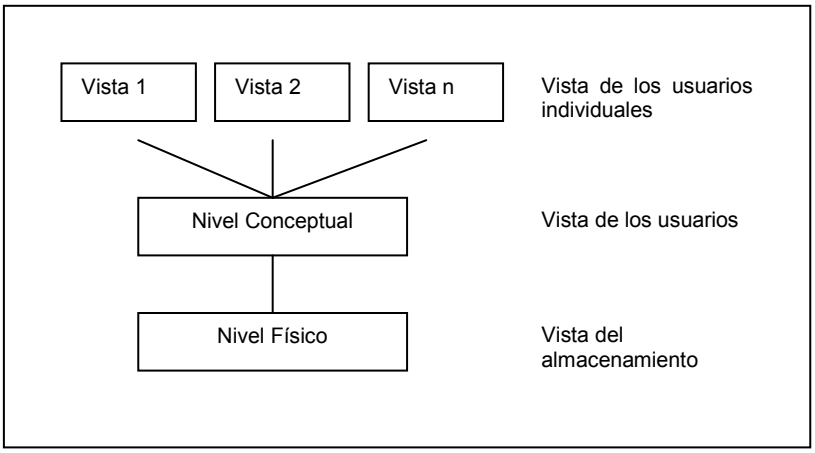


Figura 5: Niveles de abstracción

h) ***Independencia de los datos***: El esquema de una base de datos es su descripción lógica, o sea, un diagrama de los tipos de datos que se usan y sus relaciones.

El concepto de independencia de los datos se puede definir como sigue: es la capacidad de modificar una definición de esquema en un nivel, sin afectar la definición del esquema en el nivel inmediato superior.

Existen dos niveles de independencia:

- *Independencia física*: Es la capacidad de modificar el esquema físico de la base de datos sin obligar a reescribir los programas de aplicaciones.
- *Independencia lógica*: Es la capacidad de modificar el esquema conceptual sin obligar a reescribir los programas de aplicaciones.

Los beneficios que se obtienen son: hacer cambios a los programas de aplicación y modificaciones a la base de datos, con la cual, la productividad de los programadores se incrementa.

Estructura global del sistema de una base de datos

Un sistema de base de datos se encuentra integrado por varios módulos y de estos cada uno se encarga de realizar diversas tareas del sistema general. Cabe señalar que un sistema de este tipo consiste en varios componentes funcionales, destacando los siguientes como sigue (Figura 6):

1. **Manejador de archivos**: Encargado de asignar espacio en el disco y de las estructuras de datos que se van a utilizar para representar la información que será almacenada.
2. **Manejador de la Base de Datos**: Constituye la interfaz entre los datos de bajo nivel que se encuentran almacenados en la base de datos y todos aquellos programas de aplicación y las consultas que realiza el sistema.
3. **Procesador de consultas**: Este es el que traduce las proposiciones en lenguaje de consulta a instrucciones de bajo nivel que pude entender el manejador de la base de datos.
4. **Precompilador de DML**: Convierte las proposiciones en DML incrustadas en un programa de aplicaciones en llamadas normales a procedimientos en un lenguaje
5. **Compilador de DDL**: Convierte las proposiciones en DDL en un conjunto de tablas que contienen metadatos

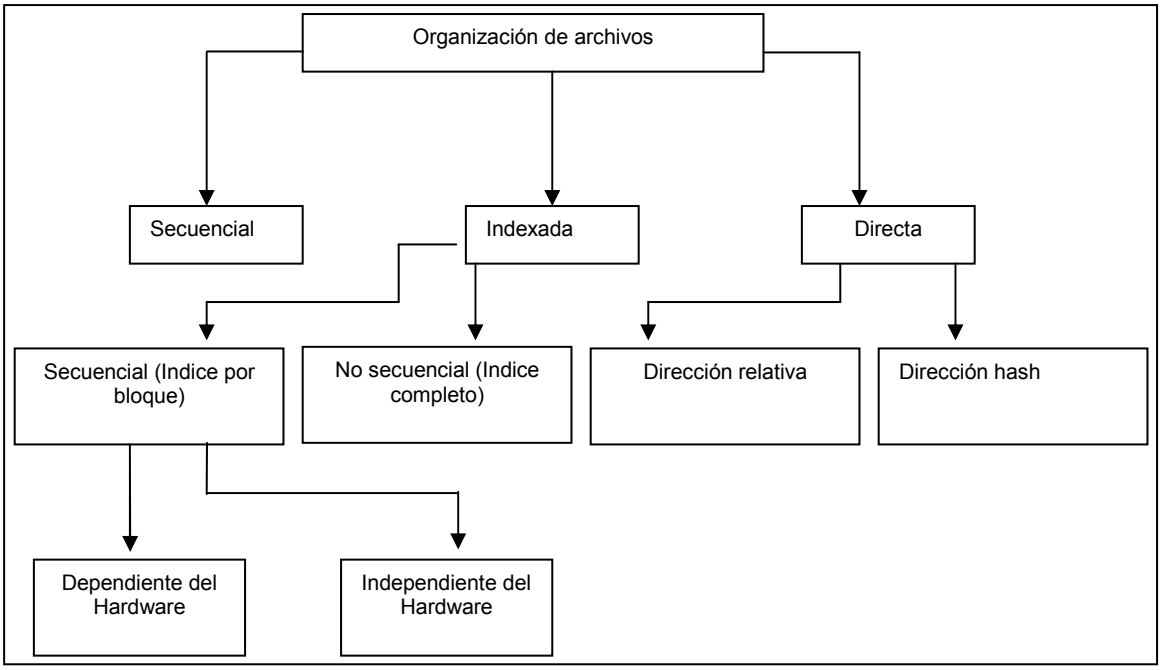


Figura 6: Organización de archivos

Diseño de una base de datos en el ciclo de vida de un sistema de información

Un sistema de información es el conjunto de actividades que regulan la distribución y el comportamiento de los datos relevantes para la administración de una empresa o sector gubernamental. Una base de datos es cualquier conjunto grande de datos que se encuentran estructurados y almacenados en una computadora. Los sistemas de gestión de base de datos (DBMS) son los paquetes de software para la gestión de dichas, en particular sirven para almacenar, manipular y recuperar datos de una computadora.

Las bases de datos son tan solo un componente de los sistemas de información, que incluyen también a los programas de aplicación, interfaces de usuario y otros tipos de paquetes de software. Cabe señalar que han sido esenciales para la supervivencia de cualquier organización ya que los datos estructurados constituyen un recurso esencial para todas los organismos (Figura 7).

El diseño de las bases de datos se encuentra situada en una perspectiva adecuada al considerarlo dentro del ciclo de vida de los sistemas de información. El diseño de un sistema de información es una actividad complicada ya que debe incluir la planificación, especificación y desarrollo de cada componente del sistema.

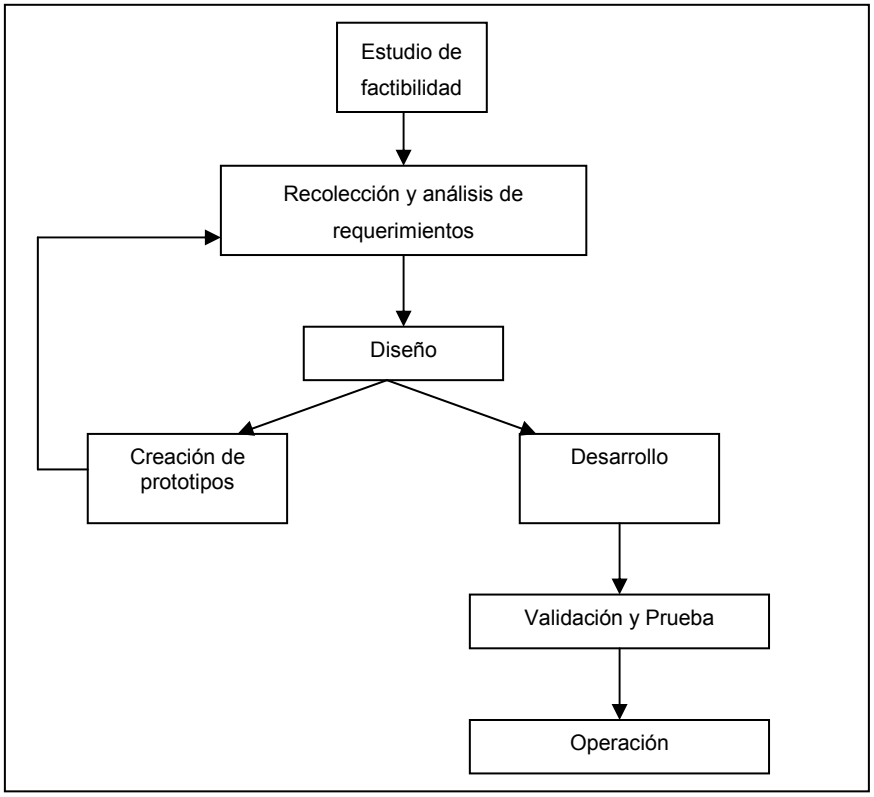


Figura 7: Ciclo de vida típica de un sistema de

Estudio de factibilidad: El estudio de factibilidad determina la rentabilidad de las distintas alternativas del sistema de información y las prioridades de los diversos componentes del sistema.

Recolección y análisis de requerimientos: La recolección de requerimientos y su análisis se ocupan de la comprensión de la misión del sistema de información, es decir, las áreas de aplicación del sistema dentro de la empresa y los problemas que debe resolver. Esta fase centra su atención en la interacción con los usuarios del sistema de información, describen sus necesidades a los diseñadores y esas descripciones se reúnen en lo que se llaman las especificaciones de requerimiento. En general, las especificaciones de requerimientos son más bien informales y desorganizadas se expresan en lenguaje natural

Diseño: Se ocupa de la especificación de la estructura del sistema de información. Se distingue entre el diseño de bases de datos y el diseño de aplicaciones. El primero es el delinear la estructura de la base de datos; el segundo es la de los programas de aplicación.

Creación de prototipos: La creación de prototipos es un añadido reciente al ciclo de vida. La mayoría de los paquetes de software incluyen actualmente herramientas para un rápido desarrollo de prototipos; con estas herramientas un diseñador puede producir con eficiencia un prototipo del sistema de información o de algunas de sus partes. Un prototipo es una realización simplificada, quizá poco eficiente, que se produce para verificar en la práctica que las fases anteriores de diseño se condujeron de forma correcta y permite a los usuarios verificar si el sistema de información satisface sus necesidades. Un prototipo que funciona es útil para la corrección o adición de requerimientos sobre la base de la experimentación.

Desarrollo: Hace referencia a la programación de la versión final y operativa de la base de datos de información.

Validación y prueba. Validación y prueba son mecanismos por los cuales se grarantiza que el buen funcionamiento que cada fase del proceso de desarrollo es de calidad aceptable y es una evolución correcta de la fase anterior.

Operación: La operación empieza con la carga inicial de datos y termina cuando el sistema se vuelve obsoleto y tiene que ser reemplazado. Durante la operación, se necesita el mantenimiento para hacer que el sistema se adapte a nuevas condiciones, mejorarlo con nuevas funciones o corregir errores no detectados durante la validación.

El ciclo de vida plantea que al diseño de bases de datos le debe anteceder el análisis de las necesidades, conduciendose en paralelo con el diseño de aplicaciones, continuando con la implementación ya sea de un prototipo o de un sistema final.

Puntos principales del diseño de un sistema de base de datos

Los sistemas para diseñar bases de datos operan sobre esquemas de datos y de funciones. Estos deben representarse y tratarse dentro del contexto de una metodología coherente para el diseño conceptual, lógico y físico de datos[2].

Es posible que el entorno automatizado que cualquier desarrollador buscaría no exista pero para el diseño de la base de datos se deben considerar las siguientes características que se desea debe poseer el sistema.

1) Interfaz amigable para el usuario

a) Debe tener una presentación coherente en pantalla y paradigmas de interacción
b) Integridad de la interfaz para usuario con el proceso de diseño
c) Permita las posibilidades de personalización

2) Amplia cobertura

a) Diagramas: Apoyo de los esquemas de datos y funciones
b) Herramientas: Conjunto completo para abarcar todo el proceso de diseño de la base de datos
c) Interfaces: Herramientas externas y sistemas de bases de datos
d) Enfoques de diseño: Descendentes, ascendentes, centrífugos y mixtos
e) Metodologías : Que tengan grados variables de imposición

3) Conjunto de herramientas robusto e integrado

a) Algoritmos rigurosos y completos
b) Orientación hacia la semántica del diseño, no sólo gráficos
c) Capacidad de analizar las implicaciones de los diseños en cuanto al rendimiento

[2] Carlo Batini – Stefano Ceri, Diseño Conceptual de Bases de Datos, pág. 467

d) Apoyo para la evaluación comparativa de diseños alternativos

e) Apoyo para esquemas malos y buenos

f) Capacidad para solucionar cuando no se tenga la información

g) Ayuda para recuperarse de resultados erróneos de las herramientas

h) Herramientas integradas en función de la arquitectura

i) Herramientas integradas funcionalmente

4) Seguimiento de la metodología y diseño

a) Definición basada en reglas de las rutas y restricciones metodológicas

b) Supervisión de la historia de diseño, de los subconjutos de diseño y de las versiones del diseño

c) Propagación de cambios

d) Acceso compartido a los diseños

5) Arquitectura abierta y extensible

a) Representaciones nuevas

b) Herramientas nuevas

c) Interfaces externas nuevas para el intercambio de información con otros sistemas

d) Metodologías nuevas

Evaluación de una base de datos

Desempeño OLTP (On Line Transaction Processing): Se ocupa del desempeño y del tiempo de respuesta del manejador de base de datos en aplicaciones en línea.

Desempeño OLAP (On Line Analytical Processing): Es una solución utilizada en el campo de la Inteligencia de Negocios, la cual consiste en consultas a estructuras multidimensionales que contienen datos resumidos de grandes Bases de Datos o Sistemas Transaccionales. La razón de usar OLAP para las consultas es la velocidad de respuesta.

Desempeño query: El desempeño y el tiempo del manejador de base de datos cuando se realizan acceso a la base de datos, para actualizar, dar de alta, baja, y modificar de registros.

Confiabilidad de número de errores: Representa el nivel de confiabilidad en los accesos a la base de datos y el manejo de los errores.

Confiabilidad de la ruta de actualización: Hace hincapié en la historia de las actualizaciones (upgrades) del producto, qué tan frecuentemente son realizadas y que tan confiables son (menos errores o bugs), otro aspecto importante es la transparencia o factibilidad para la migración hacia una actualización a otra versión del producto.

Confiabilidad de integridad y seguridad: Hace referencia a las herramientas que debe proporcionar el manejador de base de datos para asegurar la integridad de la información, el manejo de permisos y restricciones a los usuarios para respetar la confiabilidad de la información en la base de datos.

Tamaño de tablas: La tabla tiene el tamaño máximo que permite el manejador. Este punto es importante debido a que se pretende que se van a utilizar tablas que van a resguardar información y puede variar en su tamaño.

Número de usuarios: Tener contemplado él numero de usuarios que pueden acceder a la base de datos.

Multiprocesamiento: Realiza un evalúo del comportamiento de la base de datos, mientras se realizan diversas tareas al mismo tiempo o en paralelo. Se realiza una evaluación para saber cuanto puede aprovechar el hardware utilizado un CPU y que tan eficiente es la distribución de tareas.

Factibilidad de uso y aprendizaje: Se debe considerar que tan fácil es aprender y hacer uso del manejador de base de datos a corto plazo y el nivel de conocimientos que se requieren para la comprensión de esta.

Control sobre el lenguaje: Evalúa que tan poderoso es el software del manejador y el nivel de control que se tiene sobre éste para poder realizar distintas tareas en la base de datos, accesos, ajustes en el funcionamiento, modificaciones, etc.

Lenguaje de cuarta generación: Se debe considerar si este manejador proporciona herramientas de una cuarta generación por el rápido desarrollo de aplicaciones.

Interface gráfica con el usuario: Las aplicaciones se han venido realizando cada vez en forma más fáciles debido a que los usuario han desarrollado la actitud del uso de ventanas esta ventaja es importante considerarla.

Aprovechamiento de propuestas nativas del equipo: Se considera si el manejador de bases de datos aprovecha al máximo las características del hardware de un equipo y el software del sistema operativo. **Servicio:** Evalúa la calidad, eficiencia y prontitud del apoyo que proporciona el proveedor o representante.

Personal: Evalúa la cantidad de personas disponibles para el servicio, y su preparación, experiencia y habilidades.

Experiencia: Se considera el impacto, penetración y antigüedad del producto en el mercado de productos importantes de características similares, que se hayan finalizado con éxito.

Costo de consultoría: Se refiere al costo y a la calidad de la asesoría al cliente por parte del proveedor o representante.

Capacitación: Se refiere a las instalaciones, equipo, cursos, material didáctico, instructores, duración, etc., de que dispone el proveedor o representante para capacitar al cliente en el conocimiento del producto.

Instalación: Evalúa el equipo que se necesita, el tiempo requerido y la facilidad para instalar el software al cliente.

Organización: Se consideró al organismo que respalda al producto en el ámbito nacional e internacional.

Diseño de las bases de datos relacional

Modelo entidad - relación

El deseo de realizar programas con calidad la llevado a los programadores a usar modelos para construir esquemas, los cuales son representaciones de la realidad. Esta calidad de los esquemas resultantes no solo depende de la habilidad de estas personas, sino de las características del modelo de datos seleccionado. Los bloques de construcción comunes a los modelos de datos, son una colección de mecanismos de abstracción. Esta abstracción es un proceso mental que se aplica se seleccionar algunas características y propiedades de un conjunto de objetos y de estos excluir los más pertinentes.

En otras palabras, se hace abstracción al fijar atención en las propiedades consideraras esenciales de un conjunto de cosas y estudiar sus diferencias. Las abstracciones ayudan al programador a entender, clasificar y modelar la realidad. Los modelos conceptuales deben tener herramientas buenas para representar la realidad; por esta razón, deben poseer las siguientes cualidades[3]:

[3] Carlo Batini – Stefano Ceri, Diseño Conceptual de Bases de Datos, pág. 34

Expresividad: Los modelos conceptuales difieren en la elección y número de las distintas estructuras de modelado que ofrecen. En general, la disponibilidad de una amplia gama de conceptos hace posible una representación más extensa de la realidad.

Simplicidad: Un modelo conceptual tiene que ser simple, para que un esquema realizado con ese modelo sea fácil de entender por los diseñadores y usuarios de la aplicación de bases de datos. Sin embargo, la simplicidad y la expresividad son objetivos en conflicto; Si un modelo es semánticamente rico, es probable que no sea simple.

Minimalidad: Esta propiedad se consigue si cada concepto presente en el modelo tiene un significado distinto con respecto a todos los demás.

Formalidad: Los esquemas creados utilizan modelos conceptuales de datos que representan una especificación formal de la información almacenada. La formalidad requiere que todos los conceptos del modelo tengan una interpretación única, precisa y bien definida.

El modelo de datos entidad - relación (E-R) se basa a través de tener una percepción del mundo real, y que consiste de objetos básicos llamados entidades y de relaciones entre estos objetos.

Una ***entidad*** se puede definir como cualquier objeto que existe, es distinguible y se puede representar en la base de datos. Una entidad puede ser un objeto tangible o intangible. A un conjunto de entidades se le define como: un grupo de entidades del mismo tipo[4].

Las entidades se describen o representan por medio de atributos y para cada uno existe un rango de valores permitidos, llamado dominio de atributo. Formalmente, un ***atributo*** es una función que mapea un conjunto de entidades a un dominio. Así, cada entidad se describe por medio de un conjunto de parejas (atributo, valor del dato) y una pareja para cada atributo del conjunto de entidades.

Las entidades se deben diferenciar, por ello se asigna una superllave a cada conjunto. La ***superllave*** es un conjunto de uno o más atributos, que en forma conjunta permiten identificar en forma única a una entidad dentro del conjunto de entidades.

Cuando se habla de superllave el concepto no es suficiente para el modelado de la base de datos, ya que una superllave puede incluir atributos ajenos. Lo optimo que se busca es que la superllave sea lo más pequeña posible y en donde ningún subconjunto propio sea una superllave. A estas superllaves se les llama llaves candidato. El término de llave primaria se utiliza para referirse a la

[4] F. Korth Henry, Fundamentos de Bases de Datos, pág. 25

llave candidato con la cual se identificara unívocamente a las entidades dentro de un conjunto de estas. La llave primaria es el identificador de identidad formado por uno o más atributos. Cada relación tendrá combinación de atributos que tomados en conjunto, tienen la propiedad de la identificación única.

Una ***relación*** se define como una asociación entre varias entidades y un conjunto de relaciones es un grupo de relaciones del mismo tipo[5]. Estas pueden tener atributos descriptivos. La mayor parte de las relaciones en una base de datos son binarias, pero en ocasiones existen conjuntos de relaciones que incluyen a más de dos conjuntos de entidades.

Limitantes de mapeo

El modelo de entidad - relación puede definir ciertas limitantes con las que deben cumplir los datos contenidos en la base de datos. La cardinalidad de mapeo es un limitante que expresa el número de entidades con las que puede asociarse otra entidad mediante una relación.

Para ilustrar esta limitante se utilizan los conjuntos de relaciones binarias, aunque pueden existir relaciones del tipo n-arias. Para un conjunto binario de relaciones R entre los conjuntos de entidades A y B, la cardinalidad de mapeo[6] debe ser una de las siguientes (Figura 8):

- ***Una a una*: E**n donde una entidad en A está asociada con sólo una entidad B y una entidad en B está asociada con sólo una entidad A.
- ***Una a muchas*:** En donde una entidad en A está asociada con cualquier número de entidades en B, pero una entidad en B sólo puede estar asociada con una entidad en A.
- ***Muchas a una*:** Una entidad en A está relacionada únicamente con una entidad en B, pero una entidad B, puede relacionarse con cualquier número de entidades en A.
- ***Muchas a muchas*:** En donde una entidad en A está asociada con cualquier número de entidades en B y una entidad en B puede estar asociada con cualquier número de entidades en A.

Cabe señalar que las relaciones Muchas a muchas no es conveniente tener en el diseño final de la base de datos.

[5] F. Korth Henry, Fundamentos de Bases de Datos, pág. 27
[6] F. Korth Henny, Fundamentos de Bases de Datos, pág. 31

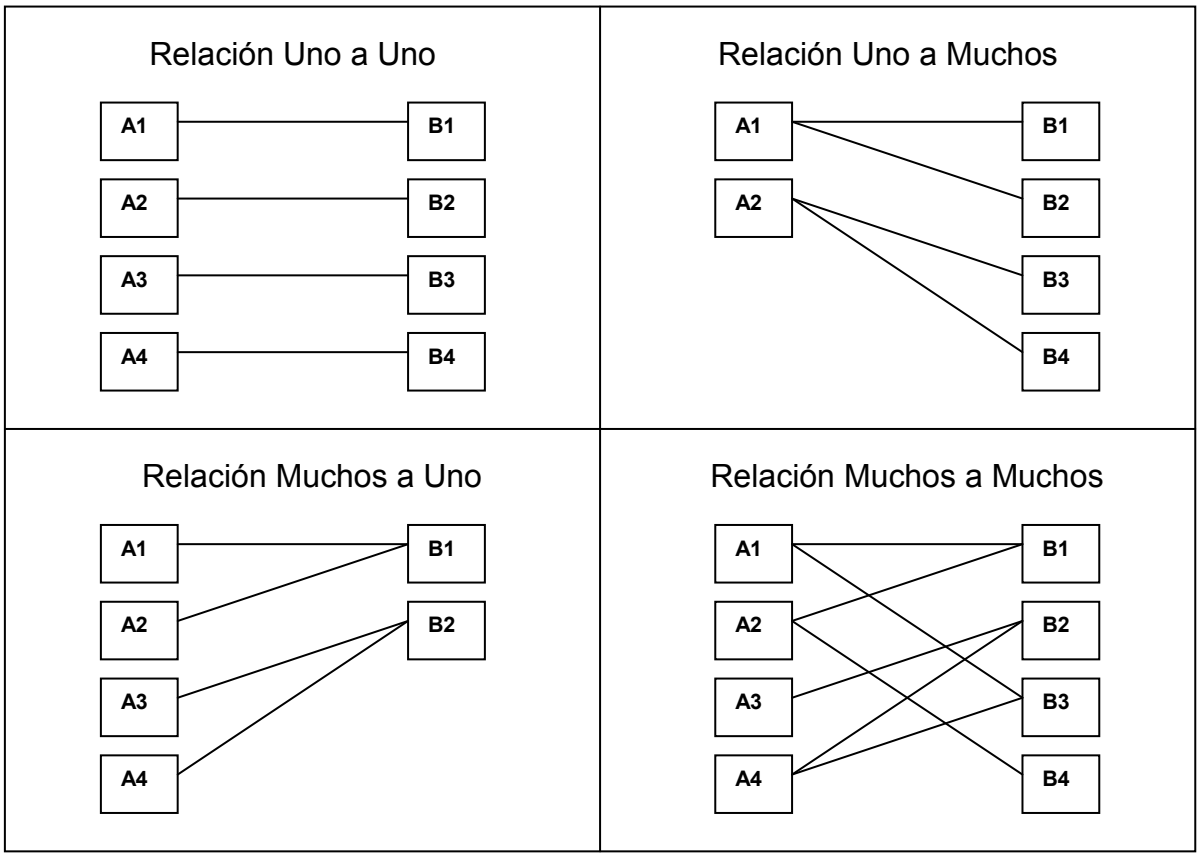


Figura 8: Cardinalidades de mapeo

Las dependencias de existencia constituyen otra clase de limitantes. Especialmente si la existencia de una entidad X de la entidad Z, se dice que x es dependiente por existencia de Z. Esto quiere decir que si se elimina Z, también se elimina X. Se dice que la entidad Z es dominante y que la entidad X es una entidad.

Esquema entidad – relación (E-R)

La estructura lógica general de una base de datos se puede expresar gráficamente mediante un diagrama entidad - relación, el cual consiste de los siguientes componentes (Figura 9):

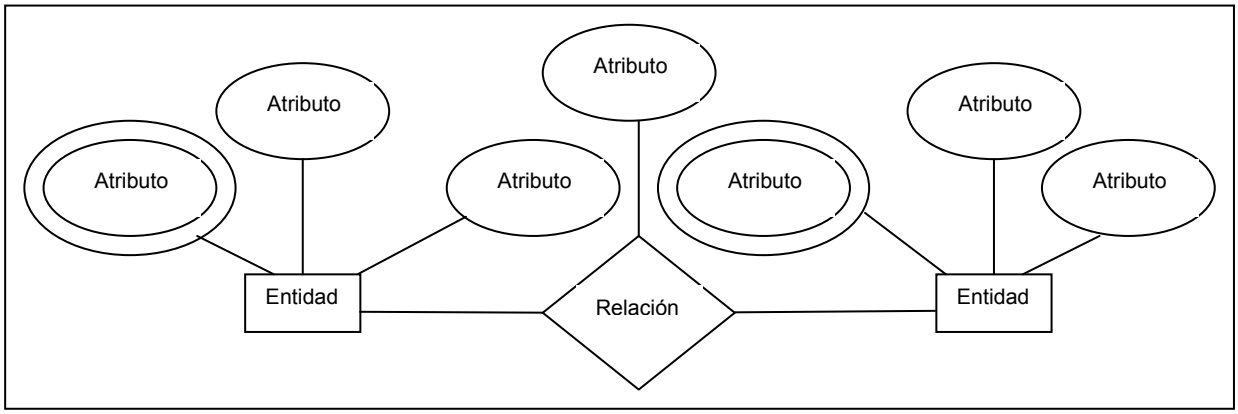
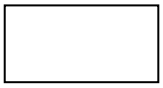
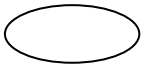
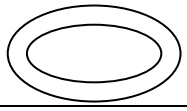
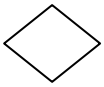


Figura 9: Diagrama de entidad relación

Los objetos que se utilizan en el diagrama relacional como sigue:

Siendo la simbología como sigue (Figura 10):

Objeto	Descripción
	Entidad: Hace referencia a un objeto que es distinguible a través de un rectángulo. Una entidad puede ser tangible o intangible.
	Atributo: Es una función, con un rango de valores permitidos, que permite mapear a un conjunto de entidades.
	Llave Primaria: Es un identificador único de una entidad
	Relación: Se refiere a la relación de dos o más entidades, puede tener atributos, permite expresar la cantidad de entidades con las que puede asociarse otra entidad.
	Conector: Es utilizado para unir a los diagramas grandes y que tienen que dividirse.
	Muchas a Muchas: Hace referencia en donde cualquier número de entidades de un mismo conjunto se encuentra asociado a otro número de entidades de otro conjunto.
	Muchas a Una: Hace referencia en donde cualquier número de entidades de un mismo conjunto se encuentra asociado a una sola y sólo una entidad.
	Una a Muchas: Hace referencia en donde una entidad se encuentra asociada a un número de entidades de un mismo conjunto

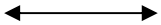
	Muchas a Muchas: Hace referencia a una relación en donde una entidad puede asociarse a otras entidades.

Figura 10: Simbología Entidad – Relación [7]

Los diagramas de una entidad relación se pueden representar también por un conjunto de tablas. Para cada conjunto de entidades y de relaciones en la base de datos, existe una tabla que recibe el nombre del conjunto de entidades o relaciones correspondientes. Cada tabla tiene un número de columnas a las cuales les corresponde un nombre único.

Perspectiva relacional

Una base de datos relacional consiste de un conjunto de tablas[7], en la que la columna de una tabla representa una relación entre un conjunto de valores. Puesto que una tabla es un conjunto de estas relaciones, existe una correspondencia entre el concepto de tabla y el concepto matemático en relación, del cual recibe nombre de modelo de datos relacional.

Las razones por la cual el modelo de datos relacional es ampliamente aceptado son porque está basado en un modelo conceptual muy fácil de entender. Este se puede resumir como sigue, una base de datos relacional consiste de tablas. Cada tabla contiene el nombre de una relación y al mismo tiempo la tabla contiene renglones y columnas.

Por lo que se define un esquema de relaciones como un conjunto de nombres de atributos para una relación y se pueden crear nuevas tablas a partir de las ya existentes. El proceso de construir nuevas tablas en el modelo relacional está gobernado por las operaciones del álgebra relacional. Además, en bases de datos relacionales, a las tablas se les denomina relaciones, a los renglones de las tablas se denominan tuplas y a las columnas se les denomina atributos.

[7] F. Korth Henry, Fundamentos de Bases de Datos, pág. 57

Base de datos relacional

Los sistemas relacionales son importantes por las peculiaridades que ofrecen en muchos tipos de proceso de datos:

- Simplicidad y generalidad
- Facilidad de uso para el usuario final
- Períodos cortos de aprendizaje
- Las peticiones de información se especifican de forma muy simple
- Independizan el diseño lógico de la implementación física

Los sistemas relacionales operan conceptualmente sobre archivos o tablas de datos (y no sobre datos individuales contenidos en el archivo), que son un medio de representar la información de una forma más compacta. Una tabla está formada por filas y columnas. Las filas de un archivo de base de datos son equivalentes a los registros de un archivo clásico (contienen los valores de los objetos o entidades descritas) y las columnas lo son a los campos (que representan los atributos de los objetos o entidades descritas).

Un concepto importante asociado a estos sistemas, es el de clave. Una clave es una referencia que se utiliza para identificar los riesgos de forma única y está formada por uno o más atributo (columnas). Se denomina clave primaria o principal a la que es mínima en cuanto al número de campos que la componen.

La clave primaria es importante, ya que permite acceder a cada uno de los elementos de la base de datos por direccionamiento asociativo mediante la combinación de tres identificadores: el nombre de la tabla, la columna y el valor de la clave.

Los sistemas de base de datos relacionales cumplen las siguientes leyes básicas:

- La tabla sólo puede contener registros con un número fijo de campos
- La base de datos, generalmente, contendrá muchas tablas
- Cada registro de la tabla es único
- El orden de los registros y el orden de los campos de la tabla no está determinado

- Por cada campo, existe un conjunto de valores posibles (dominio).

El sistema de base de datos relacional utiliza y manipula tablas que son procesadas y en donde las operaciones de acceso y modificación de los datos no requieren precisar el lugar en que se encuentran los datos.

Tipos de relaciones

Cuando se consideran las relaciones que se establezcan entre tablas, debemos poner atención y cuidado, debido a que se puede establecer un tipo de relación diferente; Las diferentes tipos de relación que se pueden establecer en una tabla son las siguientes:

- ***Uno a Uno***: Una relación uno a uno hace referencia a una situación en la que un registro de una tabla relaciona con uno sólo registro de la otra tabla.
- ***Uno a Varios***: Una relación uno a varios describe una situación en la que el elemento de información de una tabla se relaciona con varios elementos de información de otra tabla.
- ***Varios a Varios***: Una relación varios a varios escribe una situación en la que varios registros de una tabla se relacionan con varios registros de otra tabla.

Diseño de bases de datos relacionales

Un primer paso al crear una base de datos es planificar el tipo de información que se requiere almacenar en la misma teniendo que considerar dos aspectos importantes.

a) La información disponible y la información necesaria
b) La planificación de la estructura de la base de datos, en particular de las tablas, es vital para la gestión efectiva de la misma.

Así entonces crear una base de datos sin la planificación adecuada conduce, con facilidad, a tablas que tienen más o menos campos de los necesarios. Las primeras medidas que se deben tomar en cuenta antes de comenzar el diseño de una base de datos son:

1) Diseño gráfico del problema en papel (esquematizar)
2) Tomar en cuenta los datos que se van a gestionar y estimar el espacio de memoria que ocupará

Claro que los dos conceptos más importantes son: Los datos y los campos. Donde los datos constituyen la información que va contenida en la tabla. Y los diferentes tipos que se pueden almacenar como son:

- Caracteres (texto)
- Valores numéricos
- Fechas
- Informaciones lógicas (verdadero y falso)
- Imágenes

Los campos (atributo) con los distintos tipos de datos que componen la tabla. La definición de un componente requiere:

- El nombre del campo
- El tipo del campo
- La anchura del campo

La estructura de una tabla es la descripción de cada uno de los componentes que componen un registro; por tanto, uno de los principales aspectos a tener en cuenta es poder determinar el tipo de información o campos necesarios y estos deberán ser definidos en forma adecuada.

Fases del diseño de una base de datos

El diseño de una base de datos es un proceso complejo que abarca varias decisiones a muy distintos niveles. La complejidad se controla mejor si se descompone el problema en subproblemas y se resuelve cada uno de éstos independientemente, usando métodos y técnicas especificas[8] (Figura 11).

[8] Carlo Batini – Stefano Ceri, Diseño Conceptual de Bases de Datos, pág. 7,8

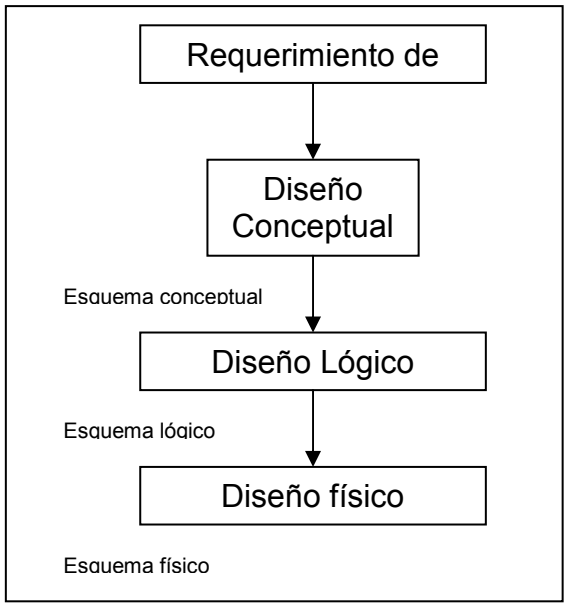


Figura 11: Enfoque orientado a los datos para el diseño de sistemas de información

- **Diseño conceptual:** El diseño conceptual es parte de la especificación de requerimientos y su resultado es un esquema conceptual de la base de datos y donde se da una descripción de alto nivel de la estructura de esta. Independientemente del software que utilice un modelo conceptual es un lenguaje que se usa para esquemas conceptuales, el propósito es describir el contenido de la información de la base de datos aún más que las estructuras de almacenamiento que se necesitaran para poder manejar la información.
- **Diseño lógico:** Este diseño es parte del esquema conceptual y da como resultado un esquema lógico, realiza una descripción de la estructura de base de datos que puede procesar el DBMS; un modelo de este tipo permite especificar esquemas lógicos; Estos modelos lógicos pertenecen a tres clases: relacional, de redes y jerárquico.
- **Diseño físico:** El diseño físico parte del esquema lógico y da como resultado un esquema físico que es una descripción de la implementación de una base de datos en la memoria secundaria. Este diseño se adapta a un sistema DBMS especifico.

Existe una retroalimentación entre el diseño físico y lógico porque las decisiones tomadas durante el primero, para mejorar el rendimiento, pueden afectar la estructura del esquema lógico. La tarea de poder determinar la forma de dividir los datos en varias tablas independientes interrelacionadas es un elemento importante del diseño de una base de datos. En ocasiones esta división es obvia,

sin embargo, en la mayoría de las ocasiones no lo es e incluso a veces no se crea el número de tablas suficientes para hacer que el sistema relacional sea óptimo y evite la redundancia de los datos. Para lograr este objetivo y obtener estructuras de datos eficientes, se utiliza una técnica llamada **normalización.**

El utilizar la normalización permite que las tablas queden optimizadas y tengan las siguientes ventajas:

- Evitar la dependencia de inserciones, actualizaciones y borrados
- Reducir la reestructuración de la base de datos debida a la introducción de nueva información
- No se establecen restricciones artificiales en la estructura de los datos

Entonces podemos decir que el diseño de la base de datos consta de tres pasos principales.

1. Definición de los datos.

Durante la primera fase del diseño de la base de datos, se realiza una lista, en papel, de todos los atributos (campos) importantes implicados en la aplicación. Para hacer esto, se examina la aplicación con detalle para determinar exactamente la clase de información que debe almacenarse en la base de datos.

2. Refinamiento de los datos.

Una vez superada la primera fase del diseño, se refina la lista inicial de campos, de modo que los campos constituyan una descripción precisa de los tipos de datos necesarios en la base de datos. En esta etapa, es vital incluir las sugerencias de tantos usuarios como sea posible, debido a que las personas que utilizan la base de datos son quienes mejor conocen la clase de información que necesitan obtener de la misma. En general, se debe asignar un campo a cualquier elemento de información que pueda utilizarse para determinar el orden de los registros o para seleccionar subconjuntos de datos.

3. Consideración de las relaciones.

En la tercera fase del diseño de una base de datos, se deben inspeccionar los campos de la tabla e intentar localizar cualquier grupo de campos repetitivo, se observa si existe algún campo que almacena la misma información una y otra vez al ir añadiendo registros (filas) a la tabla y pensar en las relaciones futuras que existirán entre los campos. Esta estrategia permitirá determinar si es conveniente o no utilizar varias tablas.

Cuando se establecen relaciones es necesario determinar un campo común, por tanto es necesario añadir un campo que identifique a los registros de forma única. El campo que establece la relación

con la clave primaria de una tabla se denomina clave ajena. Un campo clave es un campo presente en dos tablas diferentes que relaciona los registros de una tabla con los registros de otra.

En las fases del diseño es interesante hacerse algunas preguntas para determinar los datos relevantes, tales como: ¿qué clase de informes se desean de la base de datos?, ¿Qué clase de consultas se harán a esta la base de datos?

Estas consideraciones ayudan a determinar los datos que se deben almacenar y, por tanto, definir los campos que formarán la estructura de las tablas. Hay que tener mucho cuidado en la planificación de la base de datos, ya que es la fase más importante, y el tiempo empleado durante el proceso de diseño de las tablas puede ayudar a evitar muchos problemas.

Consideraciones en el diseño y la implementación

El diseño de bases de datos es un proceso de modelado, complejo y dependiente del entorno cuya finalidad es disminuir la redundancia en la información y proveer estabilidad, flexibilidad, funcionamiento aceptable y facilidad de uso al sistema de bases de datos. El proceso de diseño e implementación de bases de datos consiste en seis pasos, estos son:

1) Analizar las funciones del negocio y determinar los requerimientos de datos
2) Definir las entidades principales del sistema
3) Definir las relaciones entre entidades
4) Definir sus atributos de las entidades
5) Definir el flujo de información y la ruta de acceso de las entidades
6) Diseño físico de la base de datos

En el diseño de sistemas y la implementación de bases de datos, el término "modelo de datos" se utiliza de manera diferente. En ocasiones se hace referencia a las estructuras de datos como modelos de datos, ya sea de red, jerárquico y relacional. Sin embargo, orientado al diseño, un modelo de datos es un modelo de entidades y de sus relaciones, que son objeto de interés para una organización.

1) Analizar las funciones del negocio y determinar los requerimientos de datos

Una función de un negocio es un grupo de actividades relacionadas entre sí, que se realizan para lograr un fin dentro de una organización. Algunos métodos para recopilar información acerca de las funciones del negocio son realizar entrevistas con los usuarios, aplicar cuestionarios, revisar documentos, manuales de políticas y procedimientos, analizar sistemas existentes, observación directa y otros más.

Mediante este análisis se pretende obtener suficiente información del negocio y de los requerimientos del usuario para: definir el esquema general del negocio, delimitar las aplicaciones de los sistemas de cómputo, determinar qué procesos del negocio serán realizados por computadoras y producir información de costo/beneficio.

2) Definir las entidades principales de los sistemas

El estudio de las funciones del negocio ayuda a obtener un mayor conocimiento de una organización y permite definir las principales entidades que interesan a está. La información requerida se identifica en el análisis de los datos, que se realiza con el personal familiarizado con el entorno del negocio y se agrupa dentro de entidades que son descripciones de personas, cosas o eventos de interés para una organización.

3) Definir las relaciones entre entidades

Una vez que se han identificado las entidades de la organización, se definen las relaciones entre ellas.

Cuando se consideran las ligas entre entidades, se incrementa la información previa. Así mismo pueden existir relaciones múltiples entre las entidades, cada una representa una asociación entre las entidades por razones distintas del negocio.

4) Definir los atributos del sistema

Los elementos de datos que describen una entidad se conocen como atributos. Después de identificar las entidades y sus relaciones, se deben describir aquellas en términos de sus atributos, cual se obtiene en función de las entradas y salidas del sistema. En este punto se aplica la normalización.

5) Definir el flujo de información y la ruta de acceso de las entidades

Los puntos de acceso son los puntos de entrada a la base de datos, dependen del tipo de procesamiento y de cómo desea el usuario accesar la información.

6) Diseño físico de la base de datos

El diseño físico de bases parte del diseño lógico del sistema, obteniéndose uno propio para la implementación de un DBMS. Esta transformación la realiza el diseñador de la base de datos. Es una tarea muy técnica que requiere de amplios conocimientos y experiencia en DBMS, el proceso de diseño físico de una base de datos representa un esfuerzo continuo para proveer un desempeño aceptable y satisfacer las necesidades de los usuarios.

Normalización

La normalización es un proceso que reduce relaciones complejas a formas simples. Permite agrupar de la manera más simple posible, de tal forma que puedan hacerse cambios futuros en los requerimientos de procesamiento con un impacto mínimo en las estructuras de datos. La razón principal de partir un registro (entidad) en uno o más registros es para evitar los problemas de mantenimiento que se presentan cuando hay entidades implícitas dentro de otras entidades. El proceso de normalización se lleva acabo en las fases que se presentan a continuación.

Primera forma normal

Se pretende aislar en nuevas entidades a los atributos (o grupos de datos) que se repiten, partiendo la entidad en dos o más entidades. Los elementos que se repiten para cada tarea, son grupos repetitivos que deberían separarse de la entidad.

Habría que modificar todos los registros para ajustarlos al cambio, podrían utilizarse registros variables pero éstos requieren de procesamiento adicional. Para evitar este problema se pueden reestructurar los datos empleando la primera forma normal y aislando los grupos repetitivos.

Se observa que aislando los grupos repetitivos, se elimina el problema de modificar el tamaño de los registros. En vez de anexar campos (columnas) a los registros, previos, solamente se requiere dar de alta un nuevo registro.

Las razones por las que se usan la primera forma normal son:

- Los grupos repetitivos que son parte de un registro crean problemas. El número de ocurrencias tiende a crecer más allá de los límites originalmente esperados.

La nueva agrupación, disminuye el impacto de aumentar el tamaño de registros

- Los DBMS no son receptivos de grupos repetitivos. La solución frecuente es reservar un área grande en los registros y "esconder" los grupos repetitivos en esta área, lo que genera espacio de almacenamiento desperdiciado.
- Con frecuencia una entidad repetida posee un interés intrínseco para la compañía y su aislamiento como entidad separada es útil.

Segunda forma normal

Una vez que se han aislado los grupos repetitivos dividiendo una relación compleja, se puede simplificar aún más la relación. Para pasar a la segunda forma normal, se deben considerar los atributos que no son la llave y que son dependientes de una llave concatenada. En otras palabras, para la segunda forma normal, se debe asegurar que los atributos que no son llave de una entidad, son dependientes de todos los atributos llave de la entidad. Si no se presenta esta situación, entonces se dividen los atributos en entidades más simples.

Tercera forma normal

En la tercera forma normal hay que asegurarse de que no existen atributos que aparentemente dependen de la llave, pero que en realidad dependen de un atributo que no es la llave. A esto se le conoce como "llave escondida". Por esta razón sé aísla al atributo dentro de una nueva entidad. Si no se aíslan a las entidades, podrían presentarse anomalías de inserción, borrado y actualización, estos problemas pueden evitarse compilando la tercera forma normal, es decir, creando un nuevo archivo con una entidad separada.

Se observa que las ligas entre los archivos se establecieron a través de los atributos comunes. De esta manera, toda la información y las relaciones que forman parte del archivo sin normalizar pueden reconstruirse a través de los atributos comunes.

En algunos casos se emplea la redundancia planificada de datos con la finalidad de mejorar el desempeño del sistema.

Aunque la tercera forma normal proporciona una gran flexibilidad en el modelo de datos, con frecuencia conduce a un bajo desempeño. Algunos autores proponen una cuarta y quinta forma normal; sin embargo, el aplicar estas formas puede llevar a tener una base de datos demasiado

subdividida, de desempeño pobre e inadecuada. En la práctica, se combinan la primera y la tercera forma de tal manera que se obtiene el nivel deseado de normalización

Forma normal Boyce-Codd

Una de las formas normales más deseables que se pueden obtener es la forma normal Boyce-Codd. Un esquema de relaciones está en forma normal Boyce-Codd con respecto a un conjunto de dependencias funcionales por lo menos se cumple una de las siguientes condiciones:

- La dependencia funcional es trivial, es decir que con el atributo seleccionado se describa a los contenidos en esa entidad
- El atributo seleccionado es clave primaria del esquema de relaciones.

Un diseño de base de datos se encuentra en forma normal Boyce-Codd si cada uno de los elementos del conjunto del esquema de relación que comprende el diseño está en dicha forma normal.

Cuarta forma normal

Un esquema de relaciones en cuarta forma normal opera respecto a un conjunto de dependencias funcionales y multivaluadas si para todas las dependencias mulltivaluadas se cumple por lo menos una de las siguientes condiciones:

- La dependencia mulltivaluada es trivial
- El atributo es una clave primaria del esquema de relaciones.

Un diseño de base de datos está en cuarta forma normal si cada miembro del conjunto de esquema de relaciones que comprende el diseño está en dicha forma normal.

Análisis funcional

El análisis funcional se ocupa del modelo de un sistema de información en términos de actividades o procesos y flujos de información entre ellos. El resultado último del análisis funcional es un esquema que contiene una representación de actividades, flujos de información y otros rasgos. El esquema funcional incluye la representación de aplicaciones de bases de datos y las interacciones entre ellas[9].

[9] Carlo Batini, Stefano Ceri, Diseño Conceptual de Bases de Datos, pág. 223

Interacción entre el diseño de bases de datos y el análisis funcional

Un enfoque alternativo en el diseño de los sistemas de información. Llamado enfoque orientado a las funciones, difiere del enfoque orientado a los datos en que la atención se centra en las aplicaciones y no en los datos[10].

El análisis funcional arranca con los requerimientos de aplicaciones, descripciones de alto nivel de las actividades desarrolladas dentro de una organización y de los flujos de información entre actividades, lo cual deriva en un conjunto de esquemas que describen esas actividades y flujos de información mediante el uso de modelos de funciones específicos.

La siguiente fase del diseño funcional, llamada diseño de aplicaciones de alto nivel, transforma los esquemas de función en especificaciones de aplicación, que describen, a un alto nivel de abstracción, la conducta de los programas de aplicación, en particular, cómo las aplicaciones obtienen acceso a las bases de datos. Estas especificaciones son la base para el diseño de programas de aplicación posterior, que producen una especificación detallada de la aplicación y código de programa.

En realidad, los enfoques orientados a datos y a las funciones para el diseño de sistemas de información son complementarios; ambos aportan características positivas y se deben relacionar intimamente[1]. La idea básica de la metodología conjunta es crear el esquema conceptual de la base de datos y el esquema de funciones paralelamente, de forma que los dos procesos de diseño se influyan mutuamente. En particular, la metodología conjunta hace posible comprobar que los esquemas de datos y de funciones sean mutuamente coherentes y completos, es decir, que no generan conflictos y que todos los datos requeridos por las funciones se representen en el esquema conceptual de la base de datos y a la inversa, que las funciones incluyan todas las operaciones requeridas por la base de datos.

El diseño del esquema funcional

Una metodología para el análisis funcional debe usar la estrategia mixta para el modelado inicial de un diagrama de flujo de datos armazón a fin de descomponer el problema. Cada proceso del esquema armazón se debe entonces diseñar usando la estrategia más apropiada.

[10] Carlo Batini, Stefano Ceri, Diseño Conceptual de Bases de Datos, pág. 8,9

Puede procederse refinando los procesos hasta un nivel muy elemental. Si se consideran los procesos que en última instancia serán ejecutados por una computadora, el refinamiento podría llevarse hasta que cada proceso concuerde con un procedimiento o incluso una instrucción elemental. Obviamente, esto no llevará a un buen esquema funcional, por razones metodológicas y prácticas. En primer lugar, desde un punto de vista metodológico, el análisis debe concentrarse en definir qué hace un sistema de información, no en cómo opera dicho sistema. Por tanto, el esquema funcional resultante del análisis no debe contener aspectos procedimentales, que indican cómo se realiza un proceso, en segundo lugar, desde un punto de vista práctico, las redes demasiado complejas de procesos pequeños no son muy útiles porque no producen una visión global manejable del sistema de información. La primera consideración sugiere un criterio para terminar el análisis funcional: no seguir refinando los procesos cuando ello introduce una descripción procedimental.

Obsérvese que los flujos representan estructuras de control de lenguajes de programación convencionales (por ejemplo, interacción, ciclo, ejecución condicional).

Aspectos principales del esquema funcional

Las cualidades pueden ser difíciles de medir, guías ideales que deben conducir al diseñador a determinar el diagrama de flujo de datos más apropiado para representar los requisitos de procesamiento. Las cualidades son:

- ***Independencia funcional:*** Esta propiedad alcanza cuando cada proceso es lo bastante autónomo, es decir, que puede realizar una parte sustancial de las funciones de manera independiente. Cuando se alcanza la independencia funcional los procesos resultantes tiene las siguientes propiedades adicionales:
- ***Separabilidad***: Cada proceso puede ser analizado en detalle de manera independiente
- ***Factibilidad de integración:*** El diagrama obtenido como refinamiento de un proceso es fácil de integrar al resto del sistema
- ***Flexibilidad:***: Cada proceso es adaptable a los cambios sin crear la necesidad de modificar otros procesos
    1) ***Complexión:*** Un diagrama es completo cuando representa todos los rasgos del dominio de la aplicación con un nivel de detalle apropiado.
    2) ***Corrección:*** Un diagrama de flujo es correcto cuando usa, de forma apropiada, los conceptos del modelo de flujo de datos para representar los requerimientos.

3) ***Legibilidad:*** Un diagrama es legible cuando representa los requisitos de manera natural y puede ser entendido fácilmente sin necesidad de otras explicaciones. De la misma forma en el caso de los esquemas de datos, se distinguen las nociones de legibilidad: conceptual y gráfica.

- ***Minimalidad***: Un diagrama es mínimo cuando cada aspecto de los requerimientos aparece sólo una vez en el esquema, por ejemplo, cada actividad de la realidad debe corresponder exactamente a un proceso y los almacenes de datos no deben tener partes en común.

El diseñador puede utilizar los diagramas de flujo de datos con dos propósitos distintos:

- Producir una representación de una organización funcional existente para sobreponerle un sistema de información sin alterar las prácticas de organización y operación.
- Modelar una nueva organización de trabajo que sea más eficiente con respecto a los objetivos de la empresa.

Modelos y herramientas en el diseño de base de datos y su análisis funcional

Es muy importante hacer notar que las construcciones de un modelo también cuentan con una representación gráfica, la que permite al diseñador crear diagramas y dibujos. Estos documentos son fáciles de leer y entender, son por ello ingredientes esenciales en el proceso de diseño.

De hecho todos los modelos conceptuales se basan en el uso de unos cuantos mecanismos de abstracción y, en consecuencia, casi siempre es posible definir las correspondencias entre ellos. Particularmente, el modelo de entidad - relación surgió como la estructura formal más destacada para la representación conceptual de datos, llegando a imponerse como un estándar industrial.

De forma similar, varios modelos fueron propuestos para el análisis funcional. Estos modelos son menos homogéneos que los modelos de datos y no tan fácilmente comparables, porque el análisis funcional se aplica a problemáticas muy distintas, que van desde el procesamiento convencional de datos hasta la planificación y control en tiempo real. Surgió el modelo de flujo de datos, transformándose es un estándar de industria. Este modelo es simple y conciso, además cada elemento del modelo corresponde a un símbolo gráfico distinto.

Recientemente, atención se ha desplazado de los modelos de datos a las metodologías y herramientas de diseño. Un correcto enfoque metodológico puede ser tan importante como la elección de los modelos de datos o funciones. Además, se han desarrollado una variedad de herramientas de diseño asistidas por computadora, muchas de las cuales apoyan una representación gráfica de los esquemas de datos y funciones. Las herramientas de diseño por lo regular apoyan el modelo de entidad- relación para datos y el modelo de flujo de datos para funciones.

ACTIVIDAD I

DESARROLLO DEL SISTEMA

"DIPLOMADO INTEGRAL DE TELECOMUNICACIONES"

Objetivo: Analizar, diseñar e implementar un sistema sobre web para el pre-registro de los aspirantes al Diplomado Integral de Telecomunicaciones.

Antecedentes

El Diplomado Integral de Telecomunicaciones es un conjunto de módulos o cursos relacionados al tema de las Telecomunicaciones. El Diplomado se imparte en el Centro Educativo Multidisciplinario Polanco de la DGSCA, UNAM. Dirigido a estudiantes o profesionales relacionados al área de cómputo como una especialización para la licenciatura o para el ámbito laboral.

El sistema sobre web de pre-registro al Diplomado, se solicitó a la SSW, con el fin de agilizar la captura de la información de los aspirantes, ofrecer flexibilidad en el tiempo y lugar del registro y mostrar los datos de los aspirantes a los coordinadores oportunamente.

Definición del problema

El sistema de pre-registro consiste en que los aspirantes al Diplomado Integral de Telecomunicaciones, llenen un formulario con información requerida por la Coordinación del Diplomado, en el cual se solicitan datos como: Datos personales, Se inscribe a los cursos como, Curso al que ingresa, Nivel actual de escolaridad, ¿Por qué medio se enteró de los cursos? y Comentarios.

Investigación preliminar

En la fase de investigación preliminar, el ***usuario solicitante***[1] que solicita el sitio expone sus necesidades de un problema concreto, la SSW escucha, analiza y evalúa el problema, se cuestiona al cliente para verificar que ambos comparten la misma idea del problema y así ofrecerle las posibles soluciones, además se determina el costo del sitio o sistema sobre web.

Al cliente se le presenta un costo y plan de trabajo que comprenda las tareas y costos requeridos.
Después de acordar el costo y alguna de las posibles soluciones, se solicita la aprobación al cliente para iniciar el desarrollo del sitio.

De esta investigación se desprende que en el sitio del Diplomado Integral de Telecomunicaciones se requiere almacenar información de los aspirantes, por lo tanto, se requiere de un sistema sobre web.
El sistema sobre web del pre-registro necesita un formulario de captura para que los aspirantes registren sus datos personales, soliciten los módulos a los que desean ingresar o el ingreso al diplomado completo y consulten más información. El pre-registro no obliga a la inscripción definitiva al diplomado. Con ello se pretende conocer a los aspirantes interesados en el Diplomado Integral de Telecomunicaciones.

Requerimientos

Esta fase reúne todos los componentes que permiten tener la clara idea del problema y las herramientas óptimas con las que se desarrolla e implementa el sistema.
Para poder determinar el hardware es necesario realizar evaluaciones de transacciones de datos, almacenamiento de información, usuarios conectados, tipo de información, seguridad e integración de la comunicación.
Como el ***usuario solicitante*** pertenece a un área de docencia en la DGSCA, UNAM y la cantidad de aspirantes registrados e inscritos en el Diplomado no excede a cien solicitudes por semestre, se determinó que el sistema sería alojado en un servidor ya existente que aloja otros sistemas web de la UNAM, dado que la carga de información no es demandante.

[1] ***usuario solicitante***: Persona que hace el requerimiento a la SSW para el análisis, diseño e implementación de un sitio o sistema de información en Internet.

Requerimientos de Hardware:

a) Computadora-servidor con sistema operativo UNIX (Solaris) ó servidor LINUX.
b) Integración del servidor a un dominio en la Internet.
c) Computadora cliente con cualquier sistema operativo: Unix, Linux o Windows versión XP.
d) Conexión a Internet.

Requerimientos de Software:

a) Apache: Servidor web que permita alojar el sistema sobre web y la base de datos.
b) PHP: lenguaje de programación utilizado en el desarrollo del sistema.
c) Mysql: Manejador de base de datos.
d) SSH: Secure Shell programa de conexión remota a un servidor con sistema operativo UNIX o Linux.

Recopilación de información

El cliente envió en un documento impreso los siguientes datos que desea obtener del aspirante al Diplomado.

Datos personales

Nombre:

Fecha de nacimiento:

Dirección:

Teléfonos:

Email:

Se inscribe a los cursos como

UNAM: Estudiante | Empleado | Investigador |Académico

Otras instituciones educativas: Estudiante | Profesor | Incorporada UNAM

Particular:

Curso al que ingresa

Nombre: Diplomado: Completo o algunos módulos

Periodo: Horario:

Ha tomado cursos en esta sede: Sí | No

¿Cuántos?:

Frecuencia con que toma cursos: Frecuentemente | Casi siempre | Sólo cuando los necesito | Rara vez

Cursos que le gustaría tomar posteriormente:

Nivel actual de escolaridad

Licenciatura Carrera: Grado/Semestre:

Pasante

Titulado

Otros estudios ¿Cuál?:

Trabaja: Sí | No

Lugar:

Área:

Puesto:

¿Porque medio se enteró de los cursos?

Gaceta UNAM | Folletos | Internet | Vía telefónica | Radio | Televisión | Pizarrón | Recomendación Atención personalizada | Ya ha tomado cursos | Centro Nuevo León | Centro Mascarones | Centro Coapa | DGSCA CU | Periódico | ¿Cuál?: | Otro

Comentarios

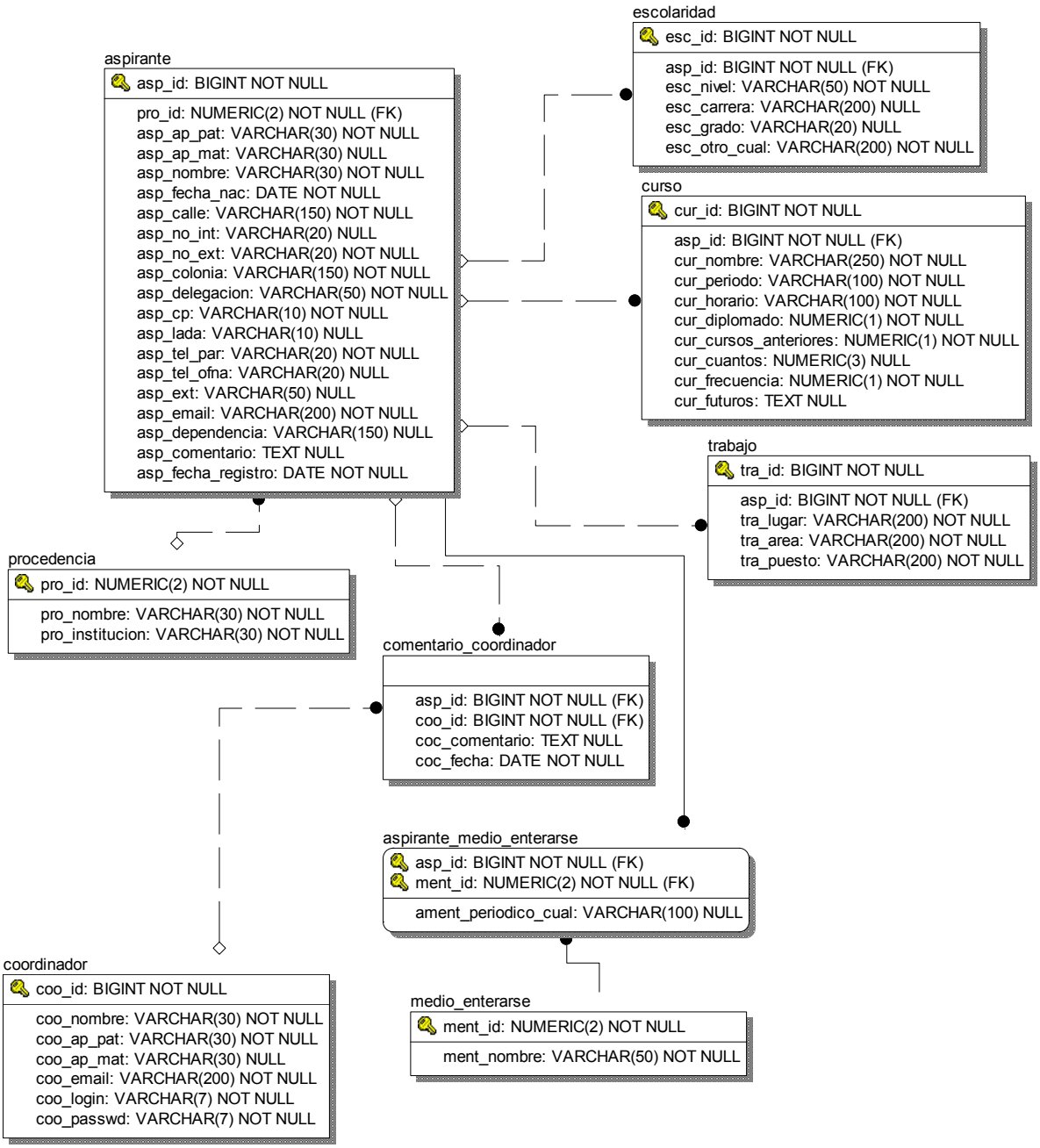
Información proporcionada por el usuario

Base de datos relacional

De acuerdo al análisis desarrollado se determinó la siguiente base de datos que durante el desarrollo del proyecto sufrió modificaciones y adecuaciones por:

1. Ajustes en interpretación
2. Longitud y tipo de datos
3. Adecuaciones al sistema

La base de datos relacional final es como se presenta a continuación y presenta las normalizaciones requeridas para la operación del sistema.



Los ajustes a la base de datos del usuario determinaron la facilidad de operación, seguridad, interpretación y manejo de información. A continuación se describen los elementos más importantes.

La información que se captura en el formulario para los aspirantes es almacenada en las tablas de la base de datos y es apoyada por varios catálogos que permiten que el manejo de la información se realice de forma confiable y segura.

Las tablas presentadas a continuación muestran su comportamiento y función en la base de datos.

Tablas en las que se almacenan los datos del aspirante incrementando automáticamente cada registro (auto increment): ***aspirante, escolaridad, curso, trabajo, procedencia, aspirante_medio_enterarse***

Tabla catálogo, en la cual los registros se almacenan previamente por medio de un script SQL para mostrarse como catálogos informativos en cajas de selección, tablas HTML o selección de checkbox y radio buttons, etc, en el sistema sobre web:
medio_enterarse

Tablas donde se almacena la información del coordinador:
coordinador
comentario_coordinador

Diseño del sistema

El diseño de un sistema indica los datos de entrada, aquéllos que serán calculados y los que deben ser almacenados. Asimismo, se escriben con todo detalle los procedimientos de cálculo y los datos individuales.

La presentación gráfica del sitio la proporciona el Departamento de Administración de Servicios Institucionales de la SSW.

Las imágenes, estilos y colores son manejados desde un punto de vista estético. Además la organización y formato de la información para la lectura de los usuarios finales en un sitio también influye en la tarea de los diseñadores.

Los componentes de diseño que integran el sitio para el Diplomado Integral de Telecomunicaciones son: un encabezado con imágenes, títulos, escudos y logos relacionados al tema, institución o servicio que ofrece el sitio y se implementan en una tabla de HTML. En el área central un menú elaborado con imágenes, texto central y vínculos. Al pie de la página imágenes, escudos y/o logos de las entidades participantes en el sitio. (ver pantalla A1.1).

Pantalla A1.1: http://www.telecompolanco.unam.mx/

Presentación del sistema para el aspirante

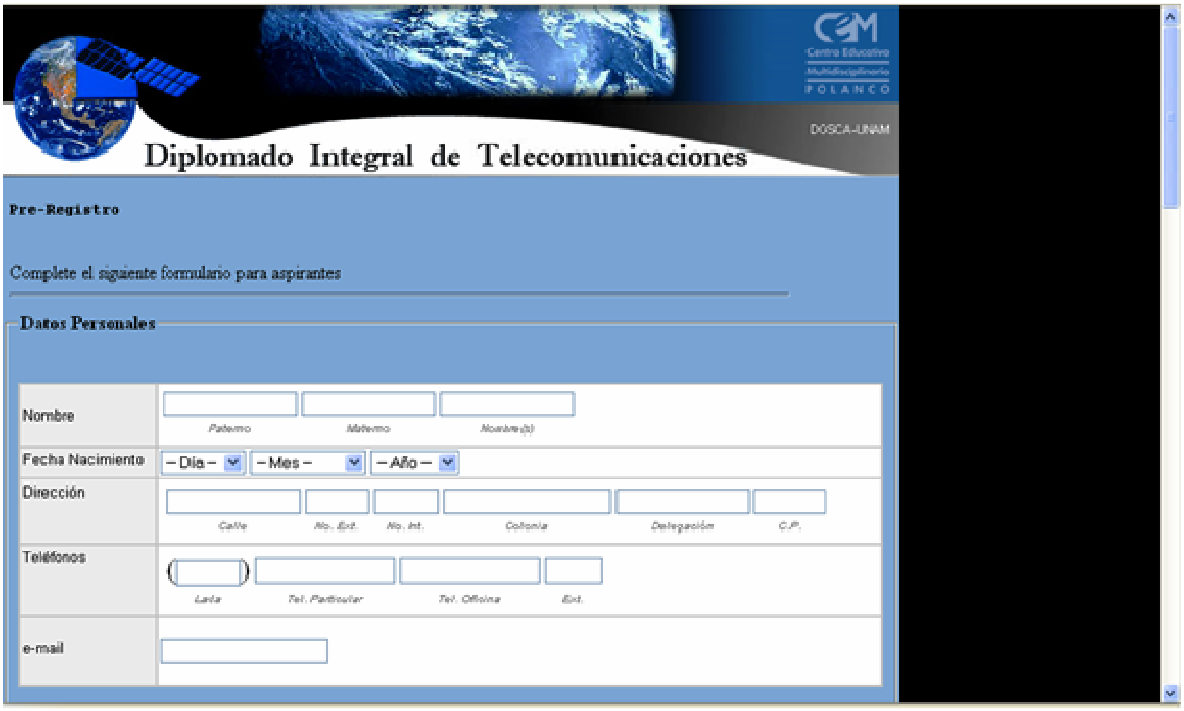
De la pantalla A.1.1, el vínculo ubicado en la parte superior del texto central Pre-registro lo enviará a la página del formulario de captura para el pre-registro de los aspirantes.

El orden y la estructura de los campos solicitados en el formulario fueron diseñados por medio de tablas en HTML, se utilizaron hojas de estilo para dar formato (tamaño, alineación, color, tipo de letra) a las celdas de las tablas y el formato (tamaño, alineación, color) de la letra en párrafos.

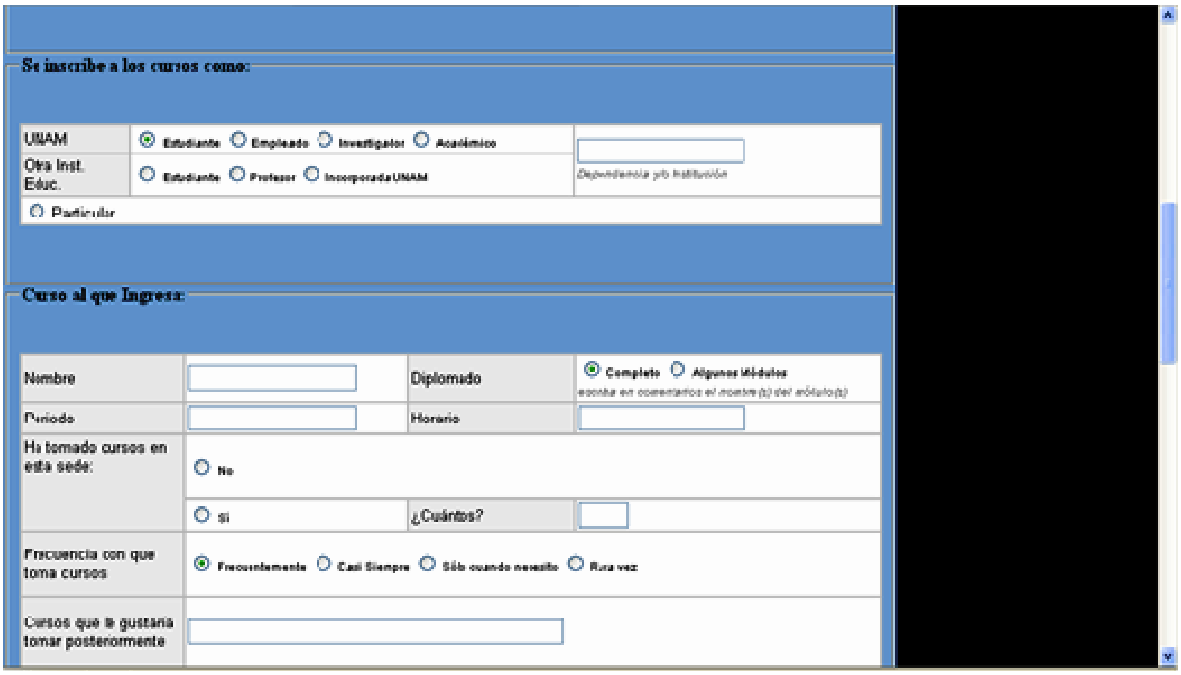
El sitio para el pre-registro de los aspirantes al Diplomado Integral de Telecomunicaciones puedes ser consultado a través de la dirección:

http://www.telecompolanco.unam.mx/pre-registro/formapre-registro.html

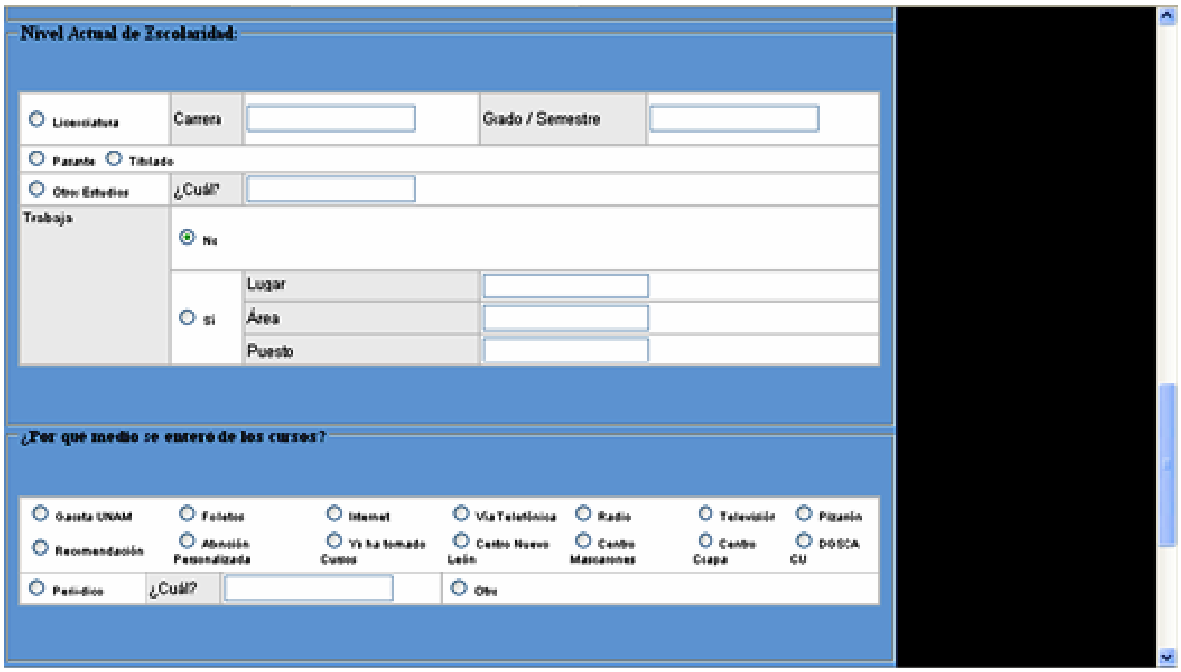
(ver pantalla A1.2), (ver pantalla A1.3), (ver pantalla A1.4), (ver pantalla A1.5).



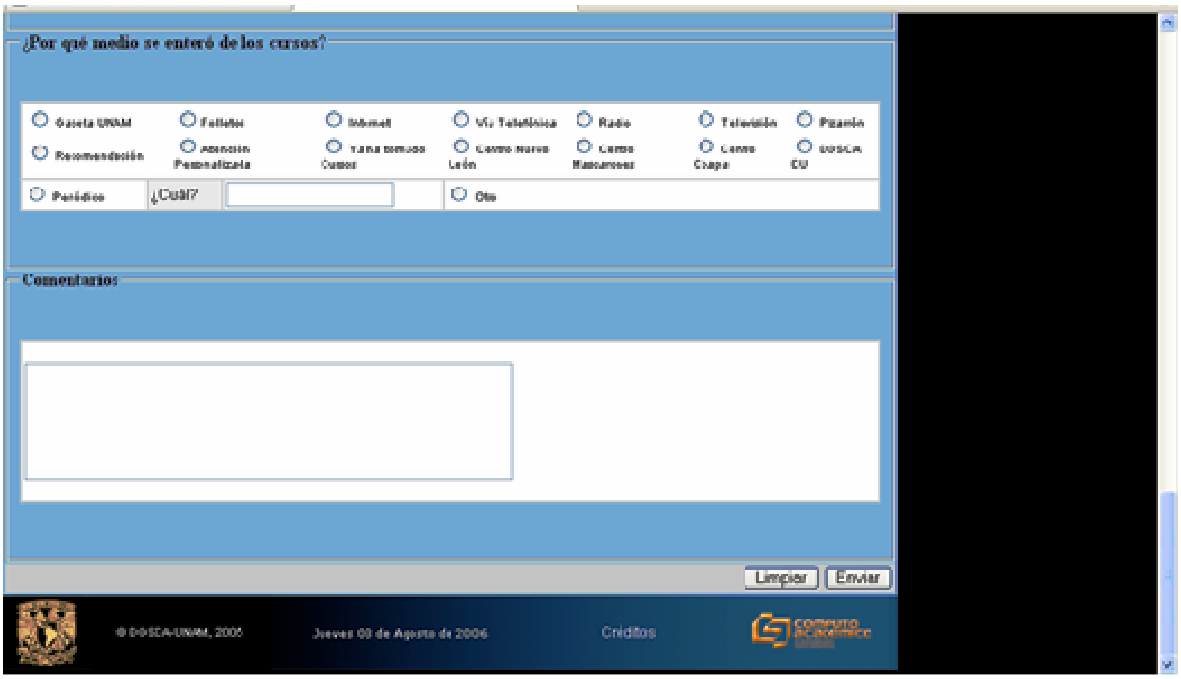
Pantalla A1.2



Pantalla A1.3



Pantalla A1.4



Pantalla A1.5

Presentación del sistema para el coordinador

El acceso a la sesión del coordinador requiere login y password por lo que la presentación y dirección del sitio será diferente al ingresar a esta página. (Ver Pantalla A1.6).
La sesión del coordinador incluye una tabla de los aspirantes registrados, con el fin de conocer a los aspirantes interesados y los comentarios o dudas que resulten para ingresar al Diplomado. La información de los aspirantes es la que previamente llenaron en el pre-registro.

Pantalla A1.6

Desarrollo del software

Como las conexiones al servidor se realizaron por medio de SSH, el manejo de los archivos de texto, imágenes y programación es desde línea de comandos del sistema operativo Linux y/o Unix.

El acceso al servidor que contiene el sistema sobre web y la base de datos es por medio de contraseñas.
El lenguaje de programación utilizado para este sistema web fue PHP y programación dinámica (Interacción únicamente con el cliente) de Javascript. El manejador de bases de datos que interactuó con PHP fue Mysql.

Los *scripts*[2] o archivos de programación fueron realizados en el editor ***vi***, por lo que las modificaciones a los archivos de programación se realizaban sobre el mismo servidor.

Los *scripts* de programación se organizaron por medio de directorios. Un directorio por tipo de usuario existente, por ejemplo: Usuario aspirante y usuario coordinador, con el objetivo de separar e identificar cada una de sus actividades. Además un directorio con el script de programación para la conexión a la base de datos y un directorio que almacena las imágenes utilizadas en el sistema.

El acceso para los aspirantes al sistema fue directamente desde la página de pre-registro, sólo el coordinador tuvo un nombre de usuario y contraseña para ingresar y hacer uso del sistema.

Programación PHP utilizada en los *scripts* para almacenar información en la base de datos. (ver cuadro I.1)

```
<?php

include("archivo.php");  //Se llama e incluye el funcionamiento de un archivo.php en el actual script que lo requiera.

$insertA = "INSERT INTO tabla VALUES(NULL,"","","","")"; //Realiza el insert a una tabla de la base de datos
$resInsert = mysql_query($insertA); //Obtiene el resultado de la acción realizar el insert
$idResIns = mysql_insert_id(); //Obtiene el id generado por el insert anterior, ese id pasa su valor a la variable $idResIns y podrá ser utilizado en inserts posteriores

?>
```

Cuadro 1

Cuadro I.1: Código Fuente

[2] *scripts*: archivos con formato .php .js .class .css que identifican el código de programación contenido de las actividades del sistema.

Programación PHP utilizada en los *scripts* para seleccionar y mostrar información en tablas de HTML desde la base de datos. (Ver Cuadro I.2)

Validación en el formato de los campos del formulario

Durante el desarrollo del sistema se validan las entradas de los campos del formulario, con el fin de enviar la información en el formato requerido y la información obligatoria que será almacenada en la base de datos. Para validar estos campos, se crean o utilizan algunas funciones en php, funciones con expresiones regulares para validar el formato de los caracteres obtenidos de los campos, es decir, si son letras o números, etc.

El script que realiza la validación de los campos en el formulario, es creado independiente del script que recibe la información del formulario, pues se tiene pensado llamar a un mismo archivo de validación en uno o varios *scripts* donde se necesiten validar campos de texto. (Ver Cuadro I.2)

```
<?php
include("validacion.php");  //Se llama e incluye el funcionamiento de un archivo.php en el actual script que lo requiera
?>
```

Cuadro I.2: Código Fuente

Ejemplo de archivo donde se realiza la función de validación en php para evitar campos vacíos "validacion.php". (Ver Cuadro I.4)

```
<?php
$info = "
      <center><input type=\"button\" value=\"Regresar\" onClick=\"history.go(-
1)\"></center>";

function validaForm($DATOSreq, $nom)
   {
    global $info;
    if($DATOSreq == "")
    {
     echo "<br><br><center><font size=\"4\" color=\"#CB1339\">No ha ingresado el campo
$nom <br><br><br>\n</font></center>".$info;
     exit;
     }
}

function Fecha($fn) // dd/mm/aaaa
   {
    $fecha = split("-",$fn);
     $dia   = $fecha[2];
    $mes   = $fecha[1];
    $anio  = $fecha[0];
```

```
if($mes>12 || $mes=="")
     return false;
   if($dia>31 || $dia =="")
     return false;
   if($anio>2020 || $anio =="")
     return false;

   if(!checkdate($mes,$dia,$anio))
   {
     return false;
   }
   else
   {
     return true;
   }
   }

function ExpRegEmail($correos)
  {
  if(!ereg("^(.+\@.+\..+)?$",$correos))
{
    return false;
  }
    else
  return true;
  }

?>
```

Cuadro I.4: Código Fuente

Ejemplo de archivo donde se recibe el valor de los campos del formulario y es llamado el archivo de "validación.php" para revisar el formato de su recepción. (Ver Cuadro I.5)

```
<?php
 include("./validacion.php");

 // Valido entradas del fomulario (nombre del campo del formulario "name= ",  nombre
que especifica la función del campo)
 // CamposVacios($DATOSreq, $nom);

  validaForm(trim($nom_paterno), "Apellido Paterno");
   validaForm(trim($nombre), "Nombre");
//Validación de fechas con la función Fecha

 $rfc_fecha = $fecha_anio.""."$fecha_mes.""."$fecha_dia;
  Fecha($rfc_fecha);

  validaForm(trim($calle), "Calle");
  validaForm(trim($num_ext), "No. Exterior");
  validaForm(trim($colonia), "Colonia");
  validaForm(trim($delegacion), "Delegacion");
   validaForm(trim($cp), "Codigo Postal");
  validaForm(trim($tel_part), "Tel. Particular");
  validaForm(trim($correo), "E-mail");


 //Validación del formato para el correo electrónico con la función ExpReg
 ExpRegEmail($correo);
?>
```

Cuadro I.5: Código Fuente

Pruebas

Las pruebas para el correcto funcionamiento del sistema de Diplomado Integral de Telecomunicaciones fueron realizadas durante y al finalizar el desarrollo.

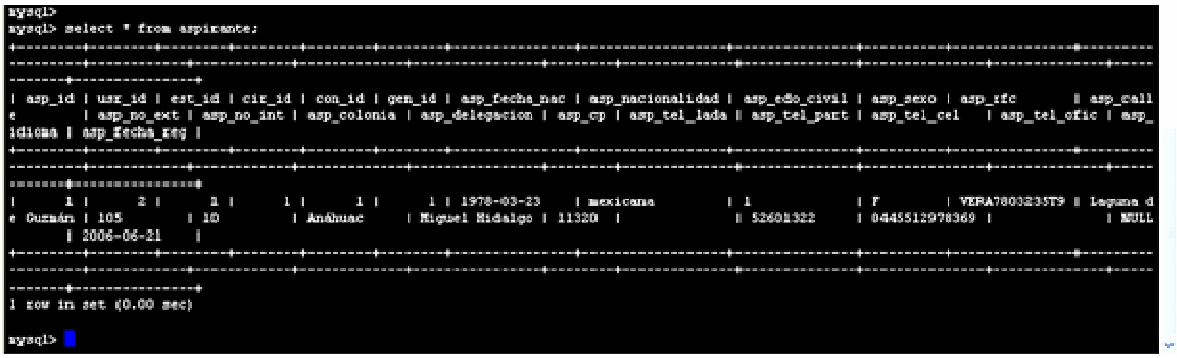
Sin embargo se verifica que el sistema realice los objetivos finales, además de validar las entradas de formularios para garantizar seguridad y asignar el formato de la información requerida que ingresará a la base de datos.

El tipo de prueba fue importante para saber exactamente cual era su respuesta con datos reales y sobre todo el comportamiento y manejo con la información.

Pruebas del diseño: Revisión de código HTML, las etiquetas de las tablas que organizan la información, imágenes, menús, las cajas de texto y componentes del formulario, el método post para el envío de datos, la ubicación de botones y vínculos.

Pruebas de validación en los campos del formulario:
Revisión de código en las validaciones por funciones en php y javascript, lo que incrementa la seguridad en el envío del formato correcto de la información y se evita la ausencia de datos en los campos obligatorios.

Pruebas de almacenamiento en base de datos:
Revisión en el código de programación para conectar a la base de datos, uso de sentencias SQL para dar de alta registros por medio de la instrucción SQL INSERT y consultas por medio de la instrucción SQL SELECT y actualizaciones con la instrucción SQL UPDATE. Durante el desarrollo del sistema se prueba directamente la instrucción SQL en la base de datos para verificar si realiza el almacenamiento de registros. (Ver Cuadro I.6)



Cuadro I.6

Pruebas para mostrar por medio de tablas HTML la información que ha sido almacenada en base de datos:

Las tablas informativas presentadas al coordinador del diplomado por medio de tablas HTML, se realizan mediante revisiones al código de programación donde la instrucción SQL SELECT que realiza la consulta de la base de datos sea la requerida, en ocasiones, la consulta solicita sólo los registros necesarios por medio de condiciones específicas.

```
$sql  = "SELECT dp.* FROM datos_personales dp, form_registro fr WHERE fr.dper_id=dp.dper_id
AND fr.freg_id=".$idReg";
$resSQL = mysql_query($sql);
```

Implementación del software

Como todos los sistemas sobre Web son desarrollados en servidores de la SSW, es necesario migrar los directorios y archivos que integran todo el proyecto del sistema desarrollado al servidor de la dependencia o institución donde será alojado.

Una forma para migrar estos archivos y directorios es por medio del programa Secure Shell SSH. Desde la línea de comandos el programa SSH realiza una conexión remota al servidor donde se migrarán todos los directorios y archivos del sistema sobre Web.

Respaldo de información y migración del sistema al servidor final

Se realizará una copia desde el servidor donde se desarrolló al servidor donde será mostrado sobre Internet. Una técnica para migrar el sistema se realiza de la siguiente manera:

1. Se comprime el directorio que integra todo el proyecto del sistema.

tar -cvf proyecto.tar proyecto //proyecto es el nombre del directorio que contiene todos los directorios y archivos de programación.

2. Se copia o envía el archivo del directorio comprimido de un servidor a otro mediante: scp archivoComprimido.tar dpolanco@cobalto.servidores.unam.mx: htdocs/

La forma para traer un archivo comprimido de un servidor externo al servidor en donde nos encontramos dentro de sesión se realiza de la siguiente manera:

scp dpolanco@cobalto.servidores.unam.mx:archivoComprimido.tar htdocs/

3. Se descomprime el archivo del proyecto del sistema en el servidor final.

tar –xvf proyecto.tar

4. Se modifica en el script de conexión el nombre de la base de datos y la contraseña en caso de ser diferentes. (Ver Cuadro I.7)

```
<?php
  //Conexion a MYSQL
  $usr    = "dpolanco";
  $passBD = "ddddddddd";
  $base   = "dpolanco";
  $idCNX  = mysql_connect("", $usr, $passBD);
//Si la conexion no se pudo realizar
  if(!$idCNX)
   {
    echo "\nProblemas al conectarse con la Base de Datos ...!\n";
    exit();
   }

  //Se establece que base de datos de usara
  mysql_select_db($base, $idCNX);
?>
```

Cuadro I.7: Código Fuente

5. Se prueba la ejecución del sistema junto con la base de datos.
6. Se realizan pruebas finales acerca del funcionamiento del sistema sobre Internet en el servidor de la dependencia a la que se le migró.

Conclusiones

La experiencia de colaborar en este proyecto me permitió integrarme a equipos de trabajo profesionales, no sólo en el sentido académico con el que me he desarrollado en la carrera de Ingeniería en Computación, sino también con un sentido de alto compromiso para resolver problemas reales a dependencias de la UNAM y sector privado. Considero muy importantes dos aspectos, el primero sobre el análisis y la evaluación de requerimientos que debe ser resuelto y el amplio sentido a la solución de problemas a través de la información obtenida del ***usuario solicitante***, de mis conocimientos y la investigación autodidacta.

ACTIVIDAD II

DESARROLLO DEL SISTEMA
"RED DE MACROUNIVERSIDADES DE AMÉRICA LATINA Y EL CARIBE"

Objetivo: Analizar, diseñar e implementar un sistema de captura para el registro de alumnos y profesores interesados en continuar con estudios de postgrado en alguna de las universidades que integran la Red de Macrouniversidades.

Áreas para estudio de postgrado: Físico-Matemáticas, Ciencias Naturales, Ciencias Sociales, etc.

Antecedentes

Con la participación de 23 representaciones institucionales, se presentó el estudio que analiza el fenómeno histórico y el desarrollo presente de las Macrouniversidades, por parte del Dr. Axel Didriksson (del CESU de la UNAM). Fue suscrita entonces la Declaración de la Ciudad Universitaria de la Red, por parte de la totalidad de las representaciones institucionales presentes.

En esta Declaración se considera como pertinente, viable y muy necesaria la Red, en la perspectiva de poner en marcha programas cooperativos y solidarios relacionados con la movilidad de estudiantes y académicos, de investigación en las fronteras del conocimiento relacionadas con la solución de los más importantes problemas de las mayorías del continente, con el postgrado, con el financiamiento público y con la preservación y el desarrollo del patrimonio histórico de estas importantes instituciones, siempre y cuando las macrouniversidades cuentan con el mayor potencial regional para realizarlos.

El programa de Becas de Movilidad Universitaria en el Postgrado, consiste en un apoyo económico para la manutención de los alumnos que realizan intercambios estudiantiles entre universidades que tengan suscritos convenios bilaterales de reciprocidad académica en la Red de Macrouniversidades de América Latina y el Caribe. La Red de Macrouniversidades Públicas de América Latina y el Caribe, y Santander Universia, consideran que:

- La movilidad universitaria constituye uno de los mecanismos fundamentales para alcanzar un mayor nivel de cooperación e integración entre las instituciones de educación superior, con el fin de fortalecer los procesos de formación y favorecer el intercambio de capacidades académicas entre los estudiantes, los profesores y los investigadores, que finalmente logre el beneficio individual y el de sus comunidades."[1]

UNIVERSIDADES QUE INTEGRAN LA RED

Universidad de Buenos Aires (Argentina)
Universidad Nacional de Córdoba (Argentina)
Universidad Nacional de la Plata (Argentina)
Universidad Mayor de San Andrés (Bolivia)
Universidad de Río de Janeiro (Brasil)
Universidad de Sao Paulo (Brasil)
Universidad de Santiago de Chile (Chile)
Universidad Nacional de Colombia (Colombia)
Universidad de Costa Rica (Costa Rica)
Universidad Nacional de Costa Rica (Costa Rica)
Universidad de Costa Rica (Costa Rica)
Universidad de la Habana (Cuba)
Universidad Central de Ecuador (Ecuador)
Universidad de El Salvador (El Salvador)
Universidad de San Carlos de Guatemala (Guatemala)
Universidad Autónoma de Honduras (Honduras)
Benemérita Universidad Autónoma de Puebla (México)
Universidad Autónoma de Sinaloa (México)
Universidad de Guadalajara (México)
Universidad Nacional Autónoma de México (México)
Universidad Nacional Autónoma de Nicaragua (Nicaragua)
Universidad de Panamá (Panamá)

[1] Antecedentes de la Red de Macrouniversidades de América Latina y el Caribe. http://www.redmacro.unam.mx/antecedentes.html

Universidad Nacional de Asunción (Paraguay)
Universidad Nacional Mayor de San Marcos (Perú)
Universidad de Puerto Rico (Puerto Rico)
Universidad Autónoma de Santo Domingo (República Dominicana)
Universidad de la República (Uruguay)
Universidad Central de Venezuela (Venezuela)
Universidad de los Andes (Venezuela)
Universidad del Zulia (Venezuela)

Investigación preliminar

El sistema para la red de Macrouniversidades de América Latina y el Caribe expone necesidades:

- Un medio de comunicación: Convocatorias, información acerca de trámites para la movilidad de alumnos, profesores e investigadores en las universidades participantes de la red, información acerca de las áreas y disciplinas impartidas en las universidades, información acerca de manutención económica para estudiantes, etc.
- Acceso personalizado: Usuario aspirante y usuario coordinador con una clave y correo para ingresar al sistema de registro.
- Formato de solicitud para aspirantes: Captura de datos personales, datos escolares, movilidad, etc, solicitada a alumnos, profesores e investigadores que desean realizar estudios de pregrado y postgrado en alguna de las universidades de la red de macrouniversidades.
- Documentos: Formatos de documentos que serán capturados en el sistema, con el fin de homologar la información solicitada por las universidades. Los documentos capturados en el sistema mostrarán la línea de trabajo que realizarán los aspirantes en los estudios solicitados.
- Unidad de coordinación: Usuario coordinador capaz de visualizar la información y documentos de los aspirantes reunida en una tabla organizada.

Durante esta fase, en equipo determinamos las funciones que requería el sistema y acordamos los siguientes aspectos: Presentar información oportuna y completa a los aspirantes mediante un sitio inteligible. Almacenar información por medio de un formato de solicitud y de documentos los cuales referimos a formatos diseñados para ser capturados en el sistema.

Los factores que involucra desarrollar un sistema sobre Web son: las funciones que realizará el sistema, si el sistema requiere sesiones para el acceso de diferentes tipos de usuario por lo que incrementará el número de tareas a realizar por el sistema, la tecnología en la que requiere ser desarrollado el sistema, el tiempo para la entrega del sistema completo y el número de desarrolladores participantes en el proyecto del sistema.

Definición del problema

Se desea construir un sistema que proporcione información oportuna y confiable de registros multimodales de la red Universidades de América Latina, además proporcione la documentación y reportes de dichos registros.

Requerimientos

Como el programa de movilidad "Red de Macrouniversidades de América Latina y el Caribe" es promovido por la UNAM y la cantidad de solicitudes desde el inicio no es demandante, aproximadamente 670 solicitudes, se determinó alojar el sistema en un servidor ya existente dentro de la DGSCA, UNAM.

Requerimientos de hardware:

a) Computadora-servidor con sistema operativo Unix (Solaris) ó servidor LINUX.
b) Integración del servidor a un dominio en Internet.

Requerimientos de software:

a) Apache: Servidor Web que permita alojar el sistema sobre Web y la base de datos.
b) PHP: lenguaje de programación utilizado en el desarrollo del sistema.
c) Mysql: Manejador de base de datos.
d) SSH: Secure Shell programa de conexión remota a un servidor con sistema operativo Linux y/o Unix.

Requerimientos **para la interfaz del registro**: Los siguientes formatos describen los datos solicitados para diseñar los formularios de captura, estos formatos fueron enviados por el *cliente*.

Número:	**Clave asignada por el sistema: 2-Clave Univ Núm Consecutivo**	2 fotografías (tamaño 2.5 X 3 cm.

PROGRAMA DE MOVILIDAD UNIVERSITARIA. NIVEL: POSGRADO

FORMATO DE SOLICITUD

Periodo de Movilidad:	INICIO dd/mm/aaaa	FIN dd/mm/aaaa

Llenar los espacios en blanco

I. DATOS PERSONALES					
Nombre:					
	Apellido Paterno	***Apellido Materno***	***Nombre(s)***		
PASAPORTE					
Dirección Actual			**Código Postal**		
Ciudad /Estado		**Teléfono Fijo**	**Código país/ciudad/**	**Teléfono**	
		Teléfono Celular			
		Correo electrónico			
Domicilio Postal					

Fecha de Nacimiento	**dd/mm/aaaa**	**Lugar de Nacimiento**	
Nacionalidad		**Sexo**	**F/M**

Llenar los espacios en blanco

II. ESTUDIOS QUE CURSA				
Institución de origen:				
Dependencia(s) que ofrece(n) el Posgrado				
Área de Conocimiento				
Posgrado	**Especifique Nombre**			
	Grado Académico	**Maestría ( )**	**Doctorado ( )**	
	Créditos Obtenidos		**Promedio**	
	Título de la tesis			
	Línea de Investigación			
	Nombre del tutor/asesor			

Llenar los espacios en blanco

III. DATOS DE LA MOVILIDAD				
Institución receptora:				
Dependencia(s) que ofrece(n) el Postgrado				
Área de Conocimiento				
Postgrado	**Especifique Nombre**			
	Grado Académico	**Maestría ( )**	**Doctorado ( )**	
Tutor que dará seguimiento:				
Dependencia de adscripción:				
Dirección Postal:				
Teléfono	**Lada**	**Teléfono Fijo**	**Celular**	
Correo electrónico				

Edad	
Sexo	
Grado académico por área de conocimiento	

Llenar los espacios en blanco

IV. BECA				
Recibe algún tipo de beca	**Si ( )**	**No ( )**	**Monto Mensual**	**Dólares**
País:			**Institución Otorgante:**	
Aspectos que cubre:				

En caso de emergencia avisar a:

Nombre:	
Parentesco:	
Dirección:	
Teléfono:	

Estoy de acuerdo con las siguientes condiciones generales para la tramitación de mi movilidad:

1. Si por algún motivo abandono el programa, notificaré de manera inmediata a la Coordinación General Regional de la Red de Macrouniversidades Públicas de América Latina y el Caribe y a Santander y o Universia de México.

2. Al término del programa de movilidad entregaré, por escrito, el informe de actividades que realicé, a mi universidad de origen y a la Coordinación General Regional de la Red de Macrouniversidades Públicas de América Latina y el Caribe. Este informe estará aprobado por el tutor que dio seguimiento a mis actividades en la universidad receptora.

3. Me comprometo a continuar inscrito y a titularme en la universidad de origen.

4. Cumpliré en su totalidad el Plan de Trabajo propuesto.

5. Tengo conocimiento de que este programa de movilidad no cubre gastos de seguro de salud.

_____ _____

Nombre y Firma del Estudiante Fecha

Vo.Bo.

Comité Académico Local de la Universidad de Origen

_____ _____

Nombre y Firma Fecha

Autoridad responsable de la Movilidad de Estudiantes

_____ _____

Nombre y Firma Fecha

Requerimientos para el documento: **Plan de trabajo**

Número:	**Clave asignada por el sistema: 2-Clave Univ-Num Consecutivo**

PROGRAMA DE MOVILIDAD UNIVERSITARIA. NIVEL: POSGRADO

FORMATO DE PLAN DE TRABAJO

Nombre:			
	Apellido Paterno	***Apellido Materno***	***Nombre(s)***
Institución receptora:			
Dependencia(s) en que realizara el proyecto			
Postrado	**Especifique Nombre**		
	Grado Académico	**Maestría ( )**	**Doctorado ( )**

OBJETIVO (S)	**ACTIVIDADES QUE SE REALIZARAN**	**CRONOGRAMA**
	Por lo menos una fila debe tener datos	

	SE AGREGAN TANTAS FILAS COMO SEAN REQUERIDAS	

Vo.Bo.

_____

Tutor de la universidad de origen

El tutor de la universidad destino deberá enviar carta de aceptación

Tutor de la Universidad de Origen Tutor de la Universidad Receptora

Recopilación de Información

La información fue analizada por un conjunto de profesionales, que apoyados por la persona experta en el manejo de información, se generaron los flujos de información y las etapas del proyecto, lo anterior con la finalidad de poder conocer desde la información inicial hasta la información de salida que debe proporcionar el sistema.

Considero que la recopilación de información es una fase muy importante porque permite obtener todos los aspectos para el diseño y es base para el desarrollo de la aplicación.

Base de Datos Relacional

De acuerdo al análisis desarrollado se determinó la siguiente base de datos que durante el desarrollo del proyecto sufrió modificaciones y adecuaciones por:

1. Ajustes en interpretación
2. Longitud y tipo de datos
3. Adecuaciones al sistema

La base de datos relacional final es como se presenta a continuación y presenta las normalizaciones requeridas para la operación del sistema.

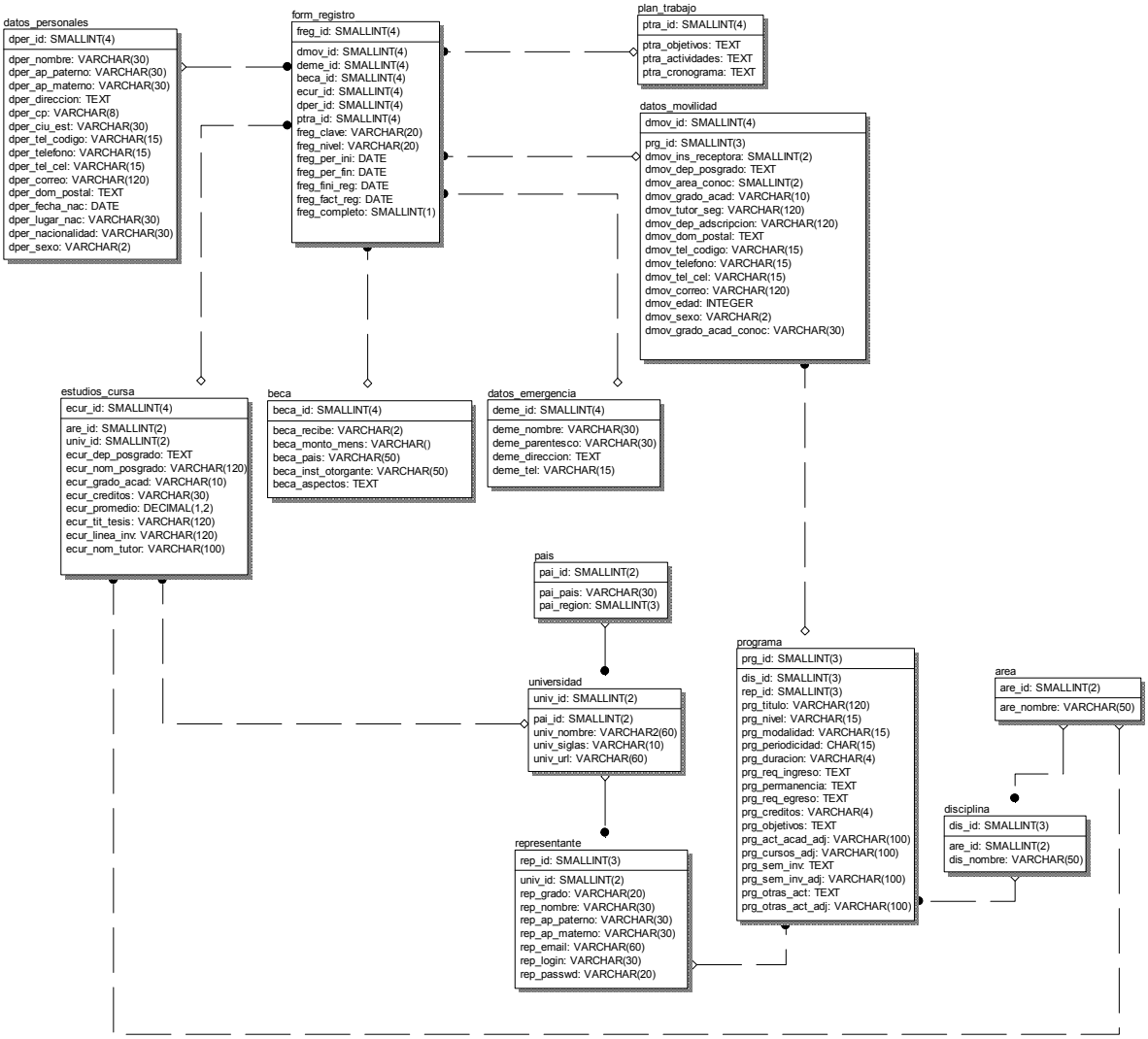


Diagrama Entidad-Relación para el sistema

"Red de Macrouniversidades de América Latina y el Caribe".

La base de datos está integrada por las tablas:

Tablas donde ingresará la información del estudiante, los registros son guardados por incremento automático con la propiedad (auto increment).

datos_personales

form_registro

plan_trabajo

datos_movilidad

estudios_cursa

beca

datos_emergencia

Tablas donde los coordinadores del sistema web, dan de alta desde el sistema web para mostrar como catálogos de selección a los estudiantes. También los registros son guardados por incremento automático.

representante

programa

Tablas catálogo, los registros son almacenados previamente por medio de scripts con instrucciones SQL que el programador ingresa a la base de datos.

país

universidad

área

disciplina

Diseño del sistema

El sitio de Red de Macrouniversidades de América Latina y el Caribe fue diseñado por el equipo del Departamento de Administración de Servicios Institucionales. El trabajo realizado en el sitio involucra el manejo de las imágenes en collage, la originalidad en la creación de logos que

muestran el concepto de la organización, los colores tanto en imágenes como en el cuerpo de la página, el menú desplegable y la distribución de la información dentro del mismo sitio.

La pantalla mostrada a continuación es la página principal o home del sitio donde visita cualquier **usuario final**[2] desde Internet con la dirección http://www.redmacro.unam.mx (ver Pantalla A2.1)

Pantalla A2.1

Interfaz mostrada a los usuarios que desean ingresar al sistema por medio de una identificación de acceso. (ver Pantalla A2.2)

[2] **usuario final**: Persona a la que se le presenta un sistema como servicio terminado para ser utilizado desde Internet.

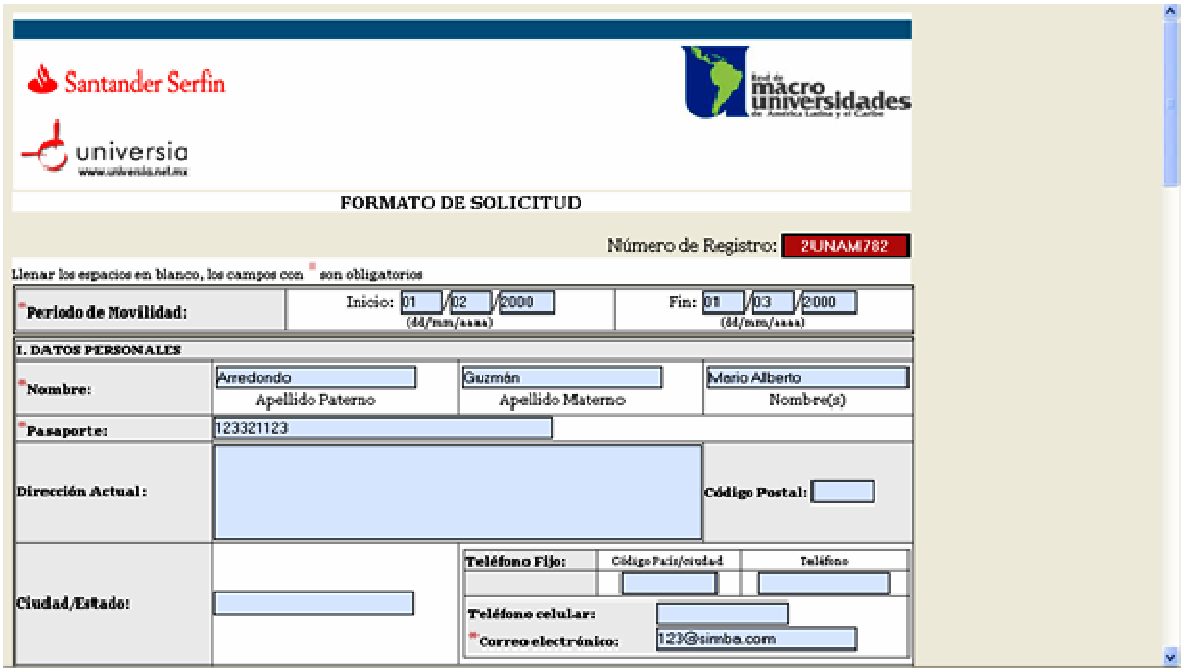
Pantalla A2.2

Si el aspirante aún no está registrado, ingresará desde:

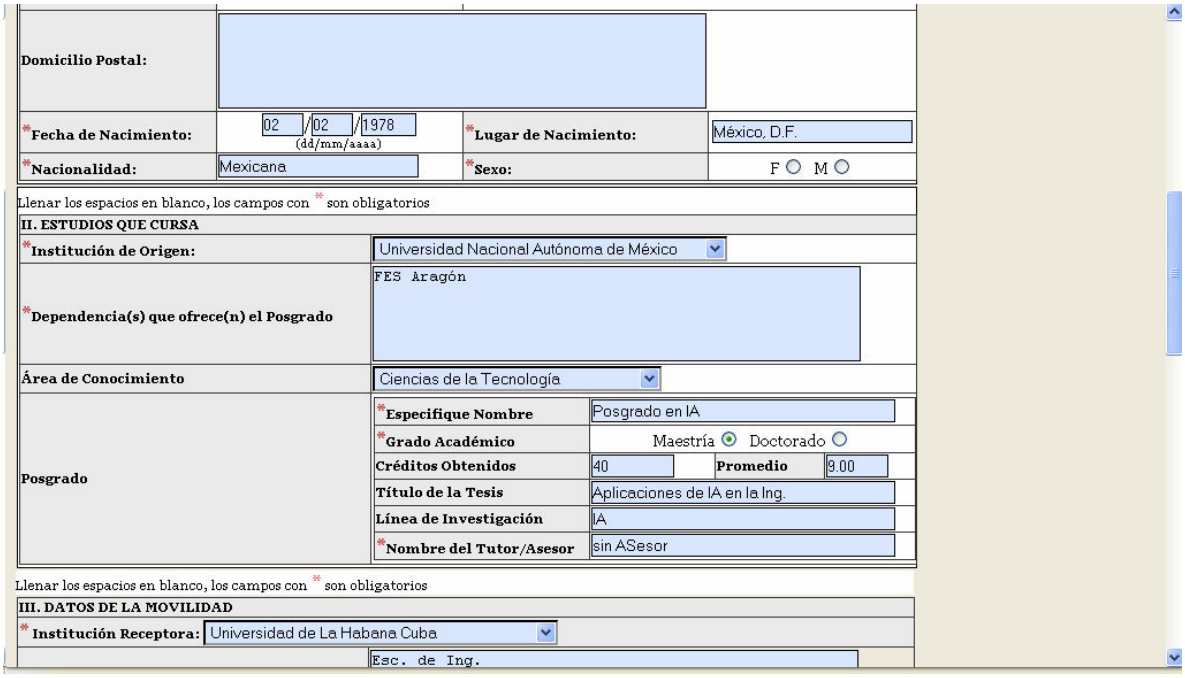
Nuevos usuarios: Registro el sistema le enviará una contraseña a su correo como clave de acceso. (ver Pantalla A2.3)

Pantalla A2.3

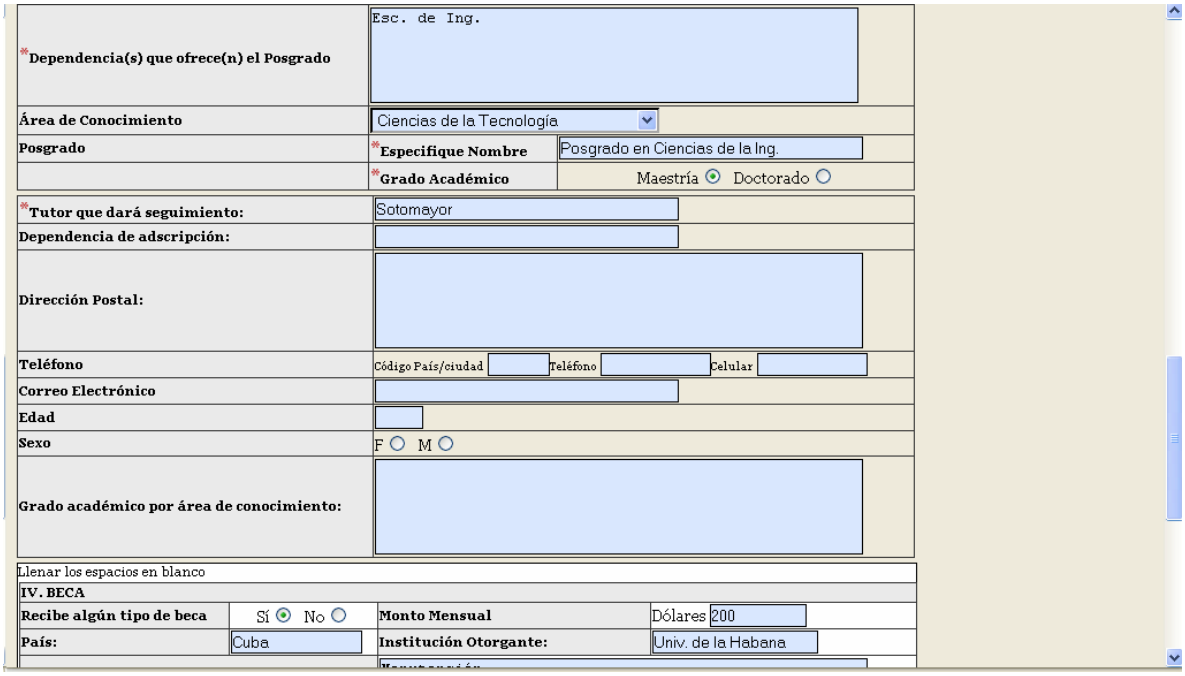
Cuando un aspirante ya fue registrado y ha ingresado su correo y clave de acceso se mostrará el siguiente Formato de solicitud. (ver Pantalla A2.4), (ver Pantalla A2.5), (ver Pantalla A2.6).



Pantalla A2.4



Pantalla A2.5



Pantalla A2.6

Interfaz presentada al Coordinador del sitio "Red de Macrouniversidades de América Latina y el Caribe". (ver Pantalla A2.7)

Pantalla A2.7

La sesión de los coordinadores del sistema "Red de Macrouniversidades de América Latina y el Caribe", contiene:

- Formulario para dar de alta un representante de área de conocimiento
- Tabla que presenta la información de los representantes por área de conocimiento y disciplina.
- Tabla que presenta la información de los aspirantes que desean realizar una movilidad de estudios de postgrado en las universidades de América Latina. Se presentan sólo aquellos que concluyen el registro y documentos solicitados.
- Información de los aspirantes que se registran y cumplen con los documentos solicitados, ésta información podrá descargar por medio de archivos txt para generar diversas estadísticas.

Desarrollo del software

El sistema fue desarrollado en el lenguaje de programación PHP, lenguaje de software libre en el que se realizan muchas aplicaciones sobre Web. Aunque PHP no es un lenguaje de programación orientada a objetos POO desde su nacimiento, se utilizó la metodología y creación de funciones para generar clases, con el fin de reutilizar código de programación en los *scripts*.

El diseño en HTML de las páginas en Internet integra validaciones dinámicas, es decir, en el correcto ingreso de datos en el formulario y manejo de cajas de selección dependientes a través de Javascript, sin embargo las validaciones también son programadas en PHP. Para asegurar que la información no perdiera su integridad debido a que el navegador en la computadora cliente pudiera tener desactivada la posibilidad de interpretar el lenguaje Javascript.

El programa SSH también fue utilizado para conectarse remotamente al servidor que contiene los proyectos de desarrollo. Manejo de archivos y directorios por medio de comandos Linux, así como el uso de editor *vi* para la edición de los archivos de programación del sistema web.

La organización de los *scripts* consiste en:

- Directorio para la creación de clases, funciones que realizan el Formato de solicitud y el documento Plan de trabajo.
- Directorio para el Formato de solicitud que contiene archivos: la hoja de estilo, el diseño del formulario en HTML, la edición del formulario.
- Directorio para el documento Plan de trabajo que contiene archivos: la hoja de estilo, el diseño del documento en HTML, la edición del documento.
- Directorio para los archivos de programación que realizan la conexión a la base de datos
- Directorio que reúne los archivos de las imágenes utilizadas dentro del sistema.

Ejemplo: código de una clase para el Registro, donde realiza altas de datos obtenidos del formulario Formato de solicitud, actualizaciones de datos del formulario, consultas de datos del aspirante registrado.
(Ver Cuadro 8)

```
<?php
include_once($rutaClases."/FormatoSolicitud.class.php");
include_once($rutaClases."/CNX.class.php");

class Registro
{
var $cnx;
  var $cnxID;
  var $llaveReg;
  var $nivel = "POSGRADO";    //En este caso sólo
posgrado
  var $clave;
  var $nivel;
  var $periodoMovilidad;      //Array
                              //$periodoMovilidad[inicio]
(dd/mm/aaaa)
                              //$periodoMovilidad[fin]
(dd/mm/aaaa)

  var $fechaInicioReg; //fecha de inicio del registro
  var $fechaUltimaAct;
  var $completo;

  var $formatoSol;    //objeto FormatoSolicitud
  var $formatoPlan;   //objeto FormatoPlan

function recuperarDatos($id_Registro)
  {
    if($id_Registro==0 || $id_Registro=="")
      return false;

    $this->cnx   = new CNX();
    $this->cnx->abrirCNX();
    $this->cnxID = $this->cnx->cnxID;

    $sql  = "SELECT * FROM registro WHERE ";
    $sql .= "reg_id=".$id_Registro;

    $resSQL = mysql_query($sql, $this->cnxID);

    if(mysql_num_rows($resSQL)<=0)
    {
```

```
      $this->clave                      = $DATOS-
>reg_clave;
      $this->periodoMovilidad[inicio]  = $DATOS-
>reg_per_ini;
      $this->periodoMovilidad[fin]     = $DATOS-
>reg_per_fin;

      $this->setFormatoSol();
      if($this->formatoSol->recuperarDatos($this-
>llaveReg))
      {
        $this->cnx->cerrarCNX();
        return true;
      }
```

Cuadro 8

Pruebas

Pruebas en el diseño: Revisión de código HTML, las etiquetas de las tablas que organizan la información, imágenes, componentes de formulario.

Pruebas de validación en los campos del formulario: Revisión de código en las validaciones por funciones en php y Javascript, lo que incrementa la seguridad en el envío del formato correcto de la información y se evita la ausencia de datos en los campos obligatorios. (Ver Cuadro 10)

```
<?php
class Validacion
{
   function validaForm($DATOSreq)
   {
    $DATOSreq=trim($DATOSreq);
    if($DATOSreq == "")
    {
    return false;
    }
    else
    return true;
   }


  function ExpRegNum($campos)
  {
   if(!ereg("^([0-9])*$",$campos))
   {
    return false;
   }
   else
   return true;
  }

  function ExpRegEmail($correos)
  {
   if(!ereg("^(.+\@.+\..+)?$",$correos))
```

```
}

 function ExpRegAlfa($texto)
 {
   if(!ereg("^([a-zA-Z0-9])*$",$texto))
   {
    return false;
   }
   else
   return true;
 }

 function ExpRegLetras($cadena)
 {
 if(!ereg("^([a-z]|[A-Z])*$",$cadena))
  {
   return false;
  }
  else
  return true;
 }

   function ExpRegPromedio($promedio)
   {
   if(!ereg("^([0-9.0-9])*$",$promedio))
    {
     return false;
    }
    else
     return true;
   }

   function Fecha($fn) // dd/mm/aaaa
   {
    $fecha = split("-",$fn);

    $dia   = $fecha[2];
    $mes   = $fecha[1];
    $anio  = $fecha[0];

    if($mes>12 || $mes=="")
      return false;
    if($dia>31 || $dia =="")
```

```
if(!checkdate($mes,$dia,$anio))
    {
      return false;
    }
    else
    {
      return true;
    }
   }
}//class
?>
```

Cuadro 10

Pruebas de almacenamiento en base de datos:

- ✓ Revisión en el código de programación para conectar a la base de datos.
- ✓ Uso de sentencias SQL para dar de alta registros por medio de INSERT, actualizaciones UPDATE y consultas por medio de SELECT.
- ✓ Durante el desarrollo del sistema se prueba directamente la sentencia SQL en la sesión de la base de datos para verificar si realiza el almacenamiento de registros.

Implementación y migración del sistema

Así como en el proyecto del sistema "Diplomado de Telecomunicaciones", la migración de archivos de un servidor a otro se realiza mediante la siguiente técnica:

1. Se comprime el directorio que integra todo el proyecto del sistema.
2. **tar -cvf proyecto.tar proyecto** //proyecto es el nombre del directorio que contiene todos los directorios y archivos de programación.
3. Se copia o envía el archivo del directorio comprimido de un servidor a otro mediante:
4. scp archivoComprimido.tar redmacro@cobalto.servidores.unam.mx: htdocs/
5. La forma para traer un archivo comprimido de un servidor externo al servidor en donde nos encontramos dentro de sesión se realiza de la siguiente manera:
6. scp redmacro@cobalto.servidores.unam.mx:archivoComprimido.tar htdocs/

7. Se descomprime el archivo del proyecto del sistema en el servidor final.
8. **tar –xvf proyecto.tar**
9. Se modifica en el script de conexión el nombre de la base de datos y la contraseña en caso de ser diferentes.

Esta migración de archivos entre servidores puede realizarse si el sistema de ambos servidores es Linux o Linux-Unix o Unix-Linux.

Alojado en el servidor final del cliente se realizan pruebas tanto en el funcionamiento del sistema como del diseño y en la conexión a la base de datos. Se crean nuevas contraseñas para los usuarios administradores que controlarán el sistema web y la base de datos.

Conclusiones

El sistema Web de "Red de Macrouniversidades de América Latina y el Caribe" me permitió conocer la responsabilidad de administrar tiempos para su desarrollo. El tiempo para la primera fase del producto fue mayor tomando en cuenta el análisis, diseño y desarrollo del sistema Web.
El sistema fue presentado en la primera convocatoria para la movilidad de estudiantes y consistió en el registro y la captura de **solicitudes** (formularios para ser llenados por los estudiantes) en el sistema para los estudiantes y para los coordinadores del programa de Redmacrouniversidades la presentación de la información de áreas de conocimientos, disciplinas, representantes y estudiantes.
La segunda fase consistió en actualizar datos requeridos en las **solicitudes**, la presentación de la información para los coordinadores y la creación de archivos txt para integrar la información de los estudiantes, información separada por un carácter especial que le permitiera al usuario abrir en programas de hoja de cálculo para la elaboración de estadísticas. La segunda fase del producto fue presentada en la segunda convocatoria y la duración fue menor, debido a que las actividades no requerían realizar nuevamente todas las etapas de la ingeniería de software.
El sistema de "Red de Macrouniversidades de América Latina y el Caribe" es un sistema visitado por miles de estudiantes, profesores e investigadores en América Latina y el mundo, por lo que incrementó la responsabilidad de desarrollar sistemas Web con mayor profesionalismo.

ACTIVIDAD III

"SISTEMA DE EVALUACIÓN DEL DESEMPEÑO DE LOS SERVICIOS ESTATALES DE EMPLEO "

SECRETARIA DE TRABAJO Y PREVISIÓN SOCIAL

Objetivo: Analizar, diseñar e implementar un sistema para la carga de información a una base de datos por medio de archivos txt.

- Desarrollar una aplicación web "CargaInformación.aspx" donde guarda archivos con extensión .txt en el sistema, archivos que contienen información de los 32 estados de la República Mexicana.
- Desarrollar una aplicación "ETL.vb" que verifique el formato requerido de la información contenida en los archivos .txt y realice cálculos con los valores que corresponden a cada estado del país. Posteriormente almacenar en base de datos, para proporcionar la información al equipo de desarrollo que realizará otras actividades dentro del sistema "Sistema de Evaluación del Desempeño de los Servicios Estatales de Empleo".

Antecedentes

La Secretaría del Trabajo y Previsión Social STPS es una dependencia gubernamental en México, su objetivo: brindar información a la ciudadanía relacionada a las condiciones generales de trabajo, condiciones de seguridad e higiene, asesoría e información técnica a empresas y trabajadores sobre las normas de trabajo.

La STPS verifica a través de 72 indicadores los servicios en los que los 32 estados de la República Mexicana tienen fallas, índices de empleo y desempleo, etc. El almacenamiento de información se enfocará principalmente a los 72 indicadores.

Definición del problema

La STPS solicita un sistema centralizado donde sea almacenada la información de 72 indicadores, sin embargo, la información no se encuentra lista para ser guardada en una base de datos, pues algunos de los datos requieren someterse a cálculos de operaciones. Además la información necesita ser recibida oportunamente desde cualquier estado de la república. La STPS necesita mostrar una tabla de cada uno de los indicadores causantes del empleo o desempleo, la tabla lista datos obtenidos por cada estado del país, por ello, organiza la información y guarda en archivos .txt.

El sistema de la STPS es muy amplio y se ha dividido en módulos de desarrollo, éste es el más importante porque es el primero que participa en el funcionamiento. Este módulo requirió ser desarrollado con mayor prioridad y ser verificado constantemente en la veracidad y confiabilidad de los resultados obtenidos por las operaciones calculadas, concretamente toda la información de los 72 indicadores que a partir de esto serían presentados en gráficas estadísticas. La responsabilidad fue mayor.

Módulos de desarrollo

- **Carga de información de indicadores al sistema y a la base de datos.**
- Presentación de información por medio de estadísticas.
- Creación de reportes sobre la información

Metodología utilizada

La tecnología de Visual Basic.Net utilizada en el desarrollo del "Sistema de Evaluación del Desempeño de los Servicios Estatales de Empleo" toma por completo la Programación Orientada a Objetos POO. La visión para el diseño y la programación se inclina en asignar cada elemento del sistema como un objeto al que se le otorgará diferentes acciones o métodos, además de otros componentes bien estructurados y creación de clases que permitan integrar las funciones similares de varios objetos para aprovechar la reutilización de código por medio de la herencia.

Visual Basic.Net de Microsoft brinda en su editor gráfico de programación un ambiente amigable y de ayuda para utilizar las clases y herramientas que facilitarán el uso adecuado durante el desarrollo del sistema.

Investigación preliminar

Los indicadores son factores de los índices de empleo y desempleo. Los archivos de cada indicador deberán ser generados previamente por los trabajadores de la STPS. La extensión de los archivos será txt. La primera columna contendrá una lista de 32 números (1-32) que identifican a los 32 estados de la república mexicana, las siguientes columnas serán "n" listas, correspondientes a "n" variables con valores numéricos y/o letras, asignadas a cada uno de los estados. Los indicadores están organizados en cinco dimensiones, por lo tanto, la ubicación de los indicadores por dimensiones facilitará reconocer la actividad de cada uno.

Dimensiones de evaluación

I. Ejercicio Presupuestal y Fortalecimiento Institucional
II. Servicios de Vinculación del Mercado de Trabajo
III. Servicios de Capacitación.
IV. Servicios de Información
V. Transversales

Los indicadores tienen un número de variables los cuáles podrán o no según sea el caso realizar cálculos y producir datos resultantes.

Dimensión III. Servicios de Capacitación

Indicador	Nombre	Descripción	Fórmula	Ponderación
C8	Cobertura de atención en diversidad de especialidades en Bécate (antes Sicat), en la modalidad Capacitación en la Práctica Laboral (antes Formación Laboral en la Práctica / Mypes).	Mide la diversidad de especialidades en la modalidad de Capacitación en la Práctica Laboral en relación al total de cursos impartidos.	Número de especialidades impartidas en la modalidad Capacitación en la Práctica Laboral **/**Total de cursos impartidos en la modalidad Capacitación en la Práctica Laboral.	2

El indicador C8 pertenece a la Dimensión III.

El formato de los archivos txt que identifican a cada uno de los 72 indicadores se muestra a continuación:

Las listas de estados y variables son separadas por medio de tabulación para poder ser leído el archivo en el sistema ETL.

El indicador C8 contiene 2 variables las cuales requieren realizar una operación de cociente y el resultado aproximarlo a la ponderación 2, es decir el máximo resultado del cociente tendrá como valor 2 y así sucesivamente ya que este sólo puede ser el máximo valor.

Número de especialidades impartidas en la modalidad Capacitación en la Práctica Laboral /Total de cursos impartidos en la modalidad Capacitación en la Práctica Laboral.

Indicador: C8 Nombre archivo: C8.txt

No. Estado	Variable1	Variable2
1	2	1
2	6052151	8214715
3	12088062	17675754
4	2966423	5397604
5	7743377	16873494
6	3388171	7506392
7	7926466	18653318
8	1322221	3574246
9	8482231	23425217
10	5257017	15009865
11	4102151	11886163
12	1386427	4220531
13	820044	2659897
14	797771	2651135
15	1766967	6307873
16	11001132	40839372
17	3873353	14639021
18	10703780	42786616
19	3853919	15531340
20	2734657	12233633
21	1494663	6853306
22	5771939	27241726
23	1161472	6886197
24	1412623	8797842
25	12003874	17365038
26	1037086	12003874
27	1880654	24809988
28	983253	17637773
29	207405	438447
30	73271	6812844
31	129840	12083209
32	13331039	

Requerimientos

Como la STPS cumple una estricta evaluación de calidad en las políticas de informática para el uso de software, el usuario solicitó las licencias para el desarrollo del sistema con Visual Studio.Net y Certificado digital de seguridad "Secure Socket Layer" (SSL) por la protección requerida en el manejo de la información.

Requerimientos de hardware:

a) Computadora con sistema operativo Windows XP.
b) Conexión a Internet.

Requerimientos de software:

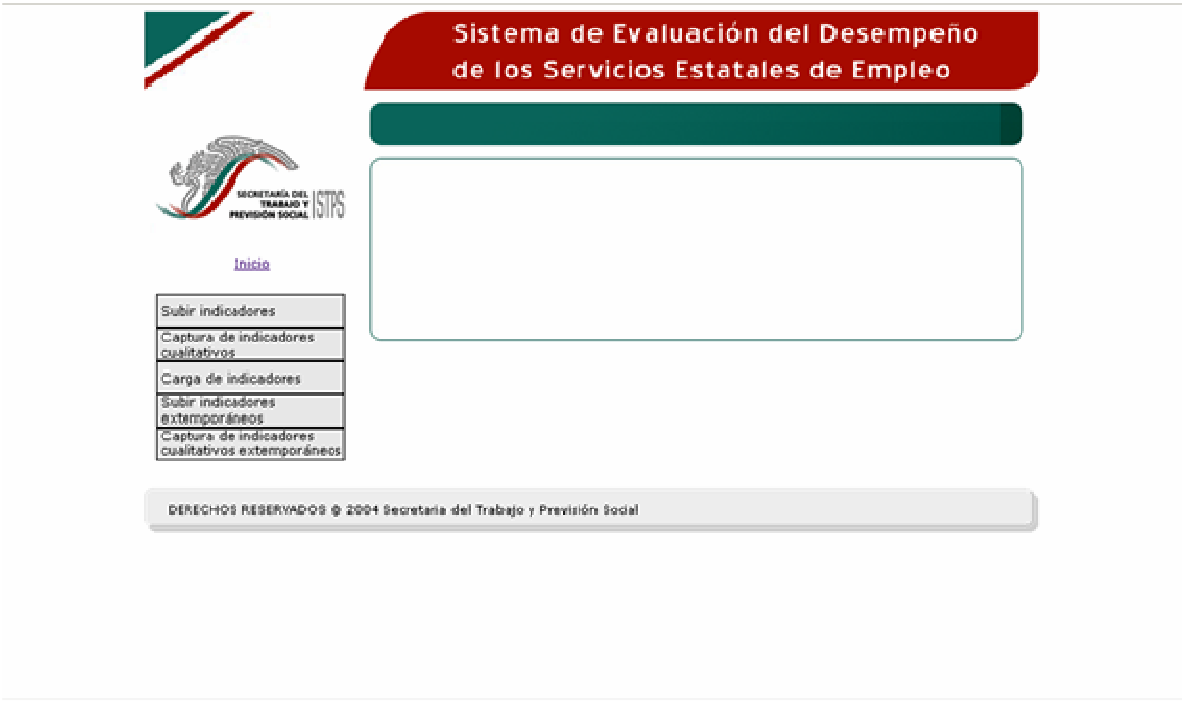
a) IIS: Servidor Web que permita alojar el sistema sobre Internet y la base de datos.
b) Visual Studio. Net versión 2003
c) Visual Basic.Net: lenguaje de programación utilizado en el desarrollo del sistema.
d) SQL Server 2000: Manejador de base de datos

Diseño del sistema

El diseño del Sistema de Evaluación del Desempeño de los Servicios Estatales de Empleo integra imágenes, colores y logos de la dependencia gubernamental Secretaria del Trabajo y Previsión Social en México, un menú que integra las funciones principales para realizar la carga de los archivos txt al sistema.

Los diseñadores muestran propuestas para seleccionar la más representativa y que cumpla con las políticas de imagen por ser una dependencia de gobierno.

La siguiente pantalla muestra el diseño que tendrá la aplicación del sistema sobre Internet. (ver PantallaA3.1)



PantallaA3.1

Desarrollo del software

El "Sistema de Evaluación del Desempeño de los Servicios Estatales de Empleo", integra dos aplicaciones realizadas en Visual Basic.Net:

Aplicación web CargaIndicadores.aspx

La aplicación es mostrada sobre Internet, lugar donde se cargaran los archivos txt de cada uno de los indicadores a un directorio del sistema. Los archivos son guardados en un directorio específico indicado en la programación. Existirá el directorio para indicadores ordinarios, que cumplen ser guardados en el periodo establecido e indicadores extraordinarios o extemporáneos que son guardados fuera del periodo establecido.

Hemos hablado acerca de guardar archivos txt, sin embargo, existe la opción de crear archivos por medio de captura, llamados indicadores cualitativos. Los indicadores cualitativos contienen 2 variables una numérica y otra alfanumérica que justificará el valor asignado a la variable numérica.

Las opciones del menú pueden ser utilizadas en un orden antes de llegar a la opción "Carga de indicadores" del mismo menú, veamos un esquema para representar la acción que genera cada una.

I

Subir indicadores
Guardar indicadores por archivo .txt

Captura de indicadores cualitativos
Guardar indicadores por medio de captura

II

Verificación de la existencia de todos los indicadores guardados, por archivo y cualitativos, antes de Cargar los indicadores al sistema

III

Carga de indicadores
Almacenamiento de todos los indicadores en el sistema correspondientes a un periodo

Prioridad de las funciones que se ejecutan en el sistema.

Explicaremos el funcionamiento de cada una de las funciones indicadas en el menú del sistema:

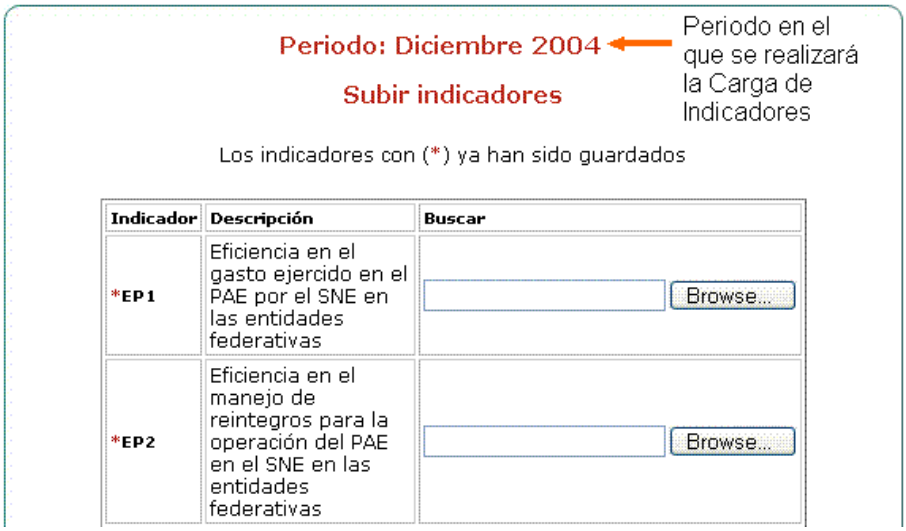
SUBIR INDICADORES POR ARCHIVO

La selección del botón "Subir Indicadores" ubicado en el menú, muestra una lista de los indicadores que podrán ser guardados por medio de archivos txt.

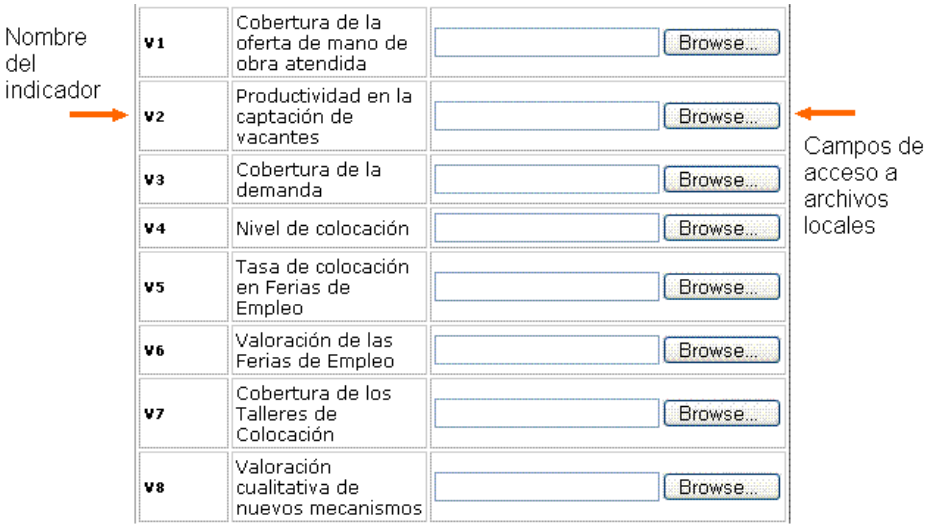
En la parte superior de la tabla se muestra el periodo al que corresponderá realizar la carga de información.

Cuando seleccione los archivos de los indicadores que desee guardar deberá verificar que corresponda el nombre del archivo al indicado en la celda de la tabla, es decir, el archivo EP1.txt será agregado en la celda del indicador EP1, con el propósito de organizar los archivos y evitar confusiones posteriores.

Al guardar un archivo éste es marcado con un asterisco rojo a un lado del nombre del indicador, en caso de volver a guardar el archivo por actualizaciones previas en su contenido automáticamente lo reemplazará. (ver Pantalla A3.2) y (ver Pantalla A3.3)



Pantalla A3.2

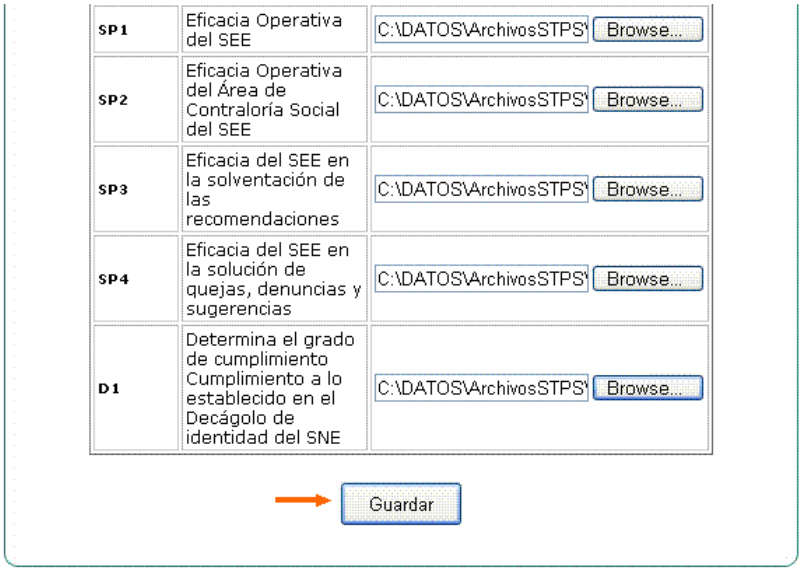


Pantalla A3.3

Desde este momento al ser guardados los archivos en el sistema no en la base de datos, el sistema valida:

- El nombre del archivo corresponda al nombre del indicador que desee subir o guardar.
- Guardar por lo menos un archivo.
- Abre el archivo y verifica las 32 líneas de los 32 estados de la República Mexicana.
- Cuenta el número de columnas correspondientes al número de variables que contiene el indicador.
- El formato de los caracteres si son numéricos o alfanuméricos.

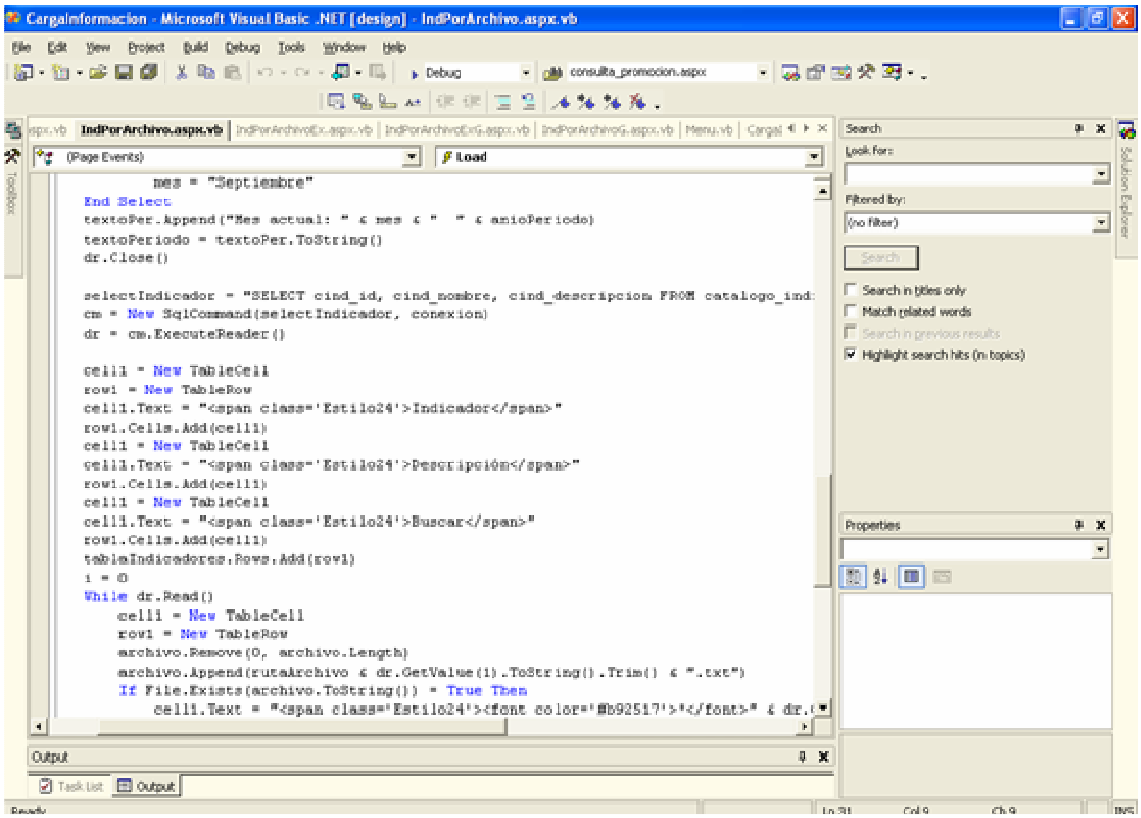
Cuando se han agregado los archivos de los indicadores en los "campos de acceso a archivos locales" podrá dirigirse al botón **Guardar,** se le confirmará si los archivos fueron guardados exitosamente de lo contrario se indicará cualquier error.
(ver Pantalla A3.4)



Pantalla A3.4

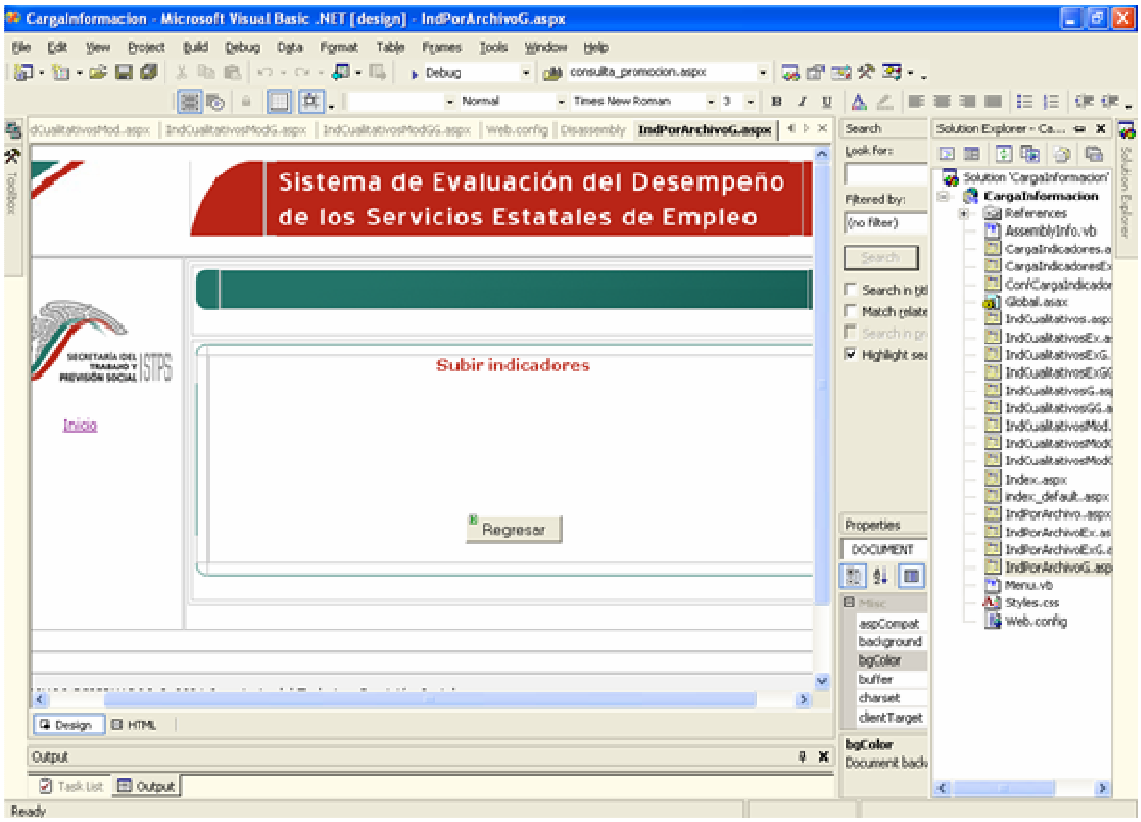
El entorno de desarrollo para la aplicación web CargaInformacion.aspx, comprende una pantalla de diseño para agregar componentes web, tablas botones, una pantalla de diseño código HTML donde podemos utilizar componentes de asp y una pantalla de programación, la cual imprime una tabla con registros resultantes de una consulta en la base de datos, proporciona funciones dinámicas en los componentes de formularios, validaciones, etc.

Por ejemplo la clase IndPorArchivo.aspx.vb muestra una tabla, la cual en la primera celda el nombre del indicador que es conocido desde la base de datos por la tabla catalogo_indicador es mostrado a través de una consulta. (ver Pantalla A3.4.a), (ver Pantalla A3.4.b) y (ver Pantalla A3.4.c),



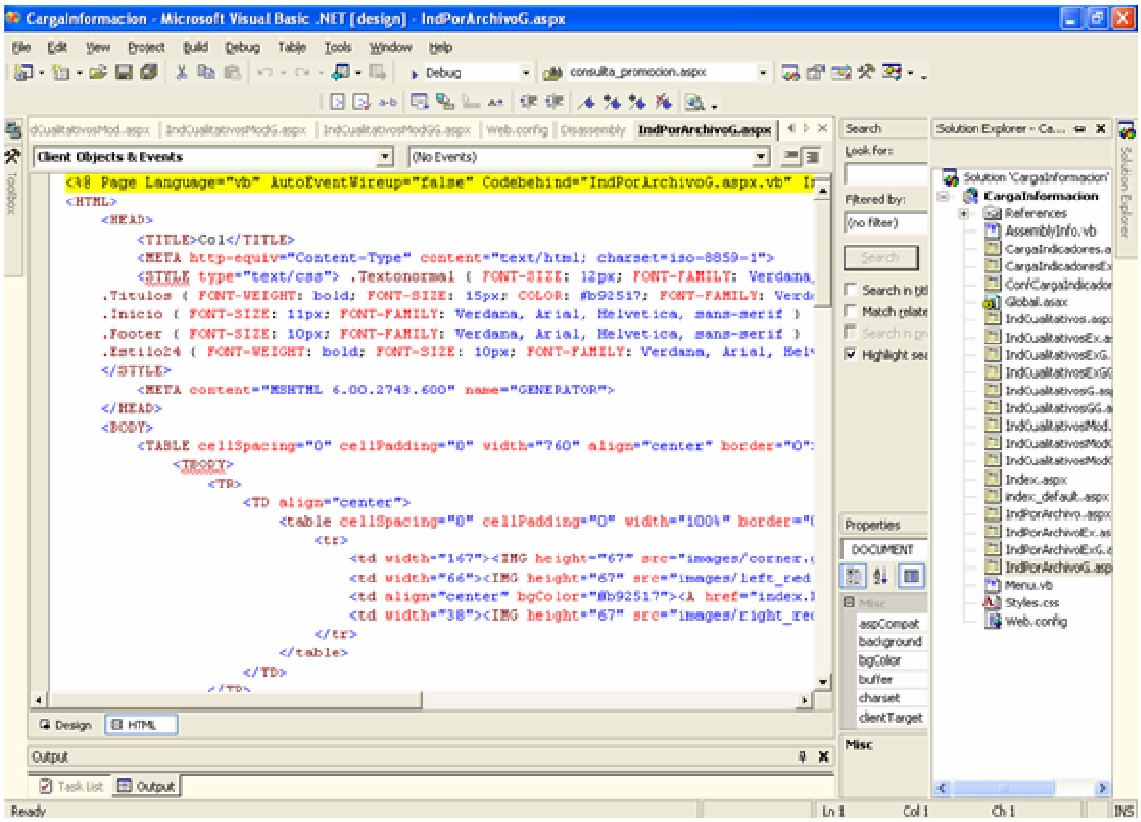
Pantalla A3.4.a

Entorno de desarrollo para la clase IndPorArchivo.aspx.bv. Código de programación para la tabla que muestra la lista de indicadores donde se subirá la información por medio de archivos .txt



Pantalla A3.4.b

Entorno de desarrollo para la clase IndPorArchivo.aspx.vb. En esta pantalla se muestra el diseño de los componentes que serán mostrados en la página. Se puede ver el botón pero la tabla no porque la tabla ha sido programada para mostrar registros obtenidos de la base de datos. En la parte derecha se lista todas las clases que han sido creadas como aplicaciones web que serán mostradas en el sistema sobre web para realizar la carga de información de indicadores por archivo, indicadores cualitativos, indicadores extemporáneos por archivo e indicadores extemporáneos cualitativos.



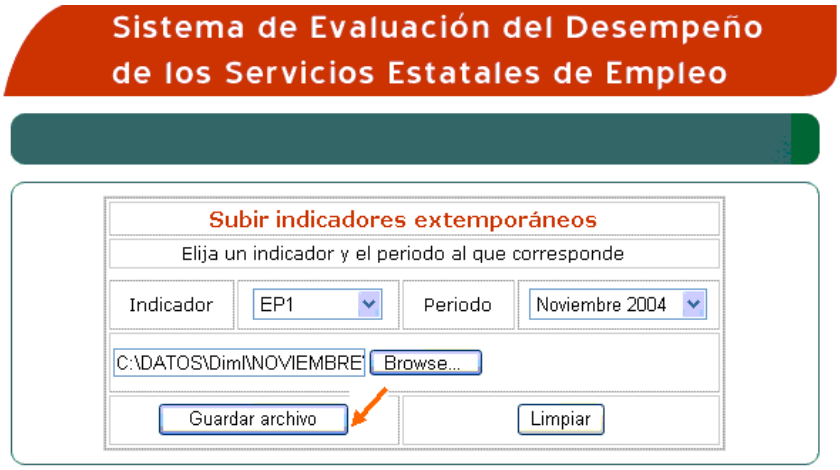
Pantalla A3.4.c

Pantalla que muestra el diseño HTML de la página que será mostrada en el sistema web, la programación se realizó en el entorno de desarrollo de la clase, sin embargo, existirán variables que requieran ser enviadas a otras páginas del sistema pasando por un <request>.

SUBIR INDICADORES EXTEMPORANEOS POR ARCHIVO

La opción de "Subir indicadores extemporáneos" será para aquellos indicadores que se encuentren fuera del periodo de carga que les corresponde, periodos anteriores principalmente, por ello deberá elegirse el nombre del indicador y el periodo al que será guardado.

Únicamente estos archivos podrán guardarse uno a uno, a diferencia de la lista mostrada en "Subir indicadores", que son indicadores ordinarios. (ver Pantalla A3.5)



Pantalla A3.5

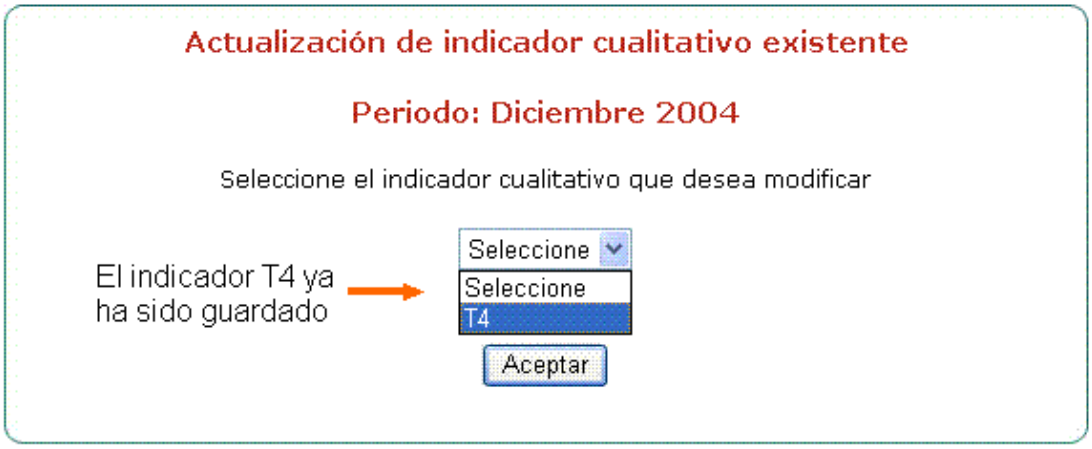
CAPTURA DE INDICADORES CUALITATIVOS

Esta opción permite crear archivos txt para los indicadores cualitativos por medio de captura. (ver Pantalla A3.6)

- Deberá seleccionar el indicador que desee crear, la caja de selección contiene el nombre de los indicadores cualitativos que son conocidos desde la base de datos.
- Llenar los campos de valor y justificación en los estados que así lo requieran.

En la parte superior se encuentra una liga que le permitirá modificar un indicador cualitativo sólo si éste existiera, es muy recomendable para aquellos indicadores que necesiten actualizaciones o cambios en algunos estados no todos, ya que permitirá mostrar el contenido del archivo anteriormente guardado y realizar ajustes mínimos. (ver Pantalla A3.6.1)

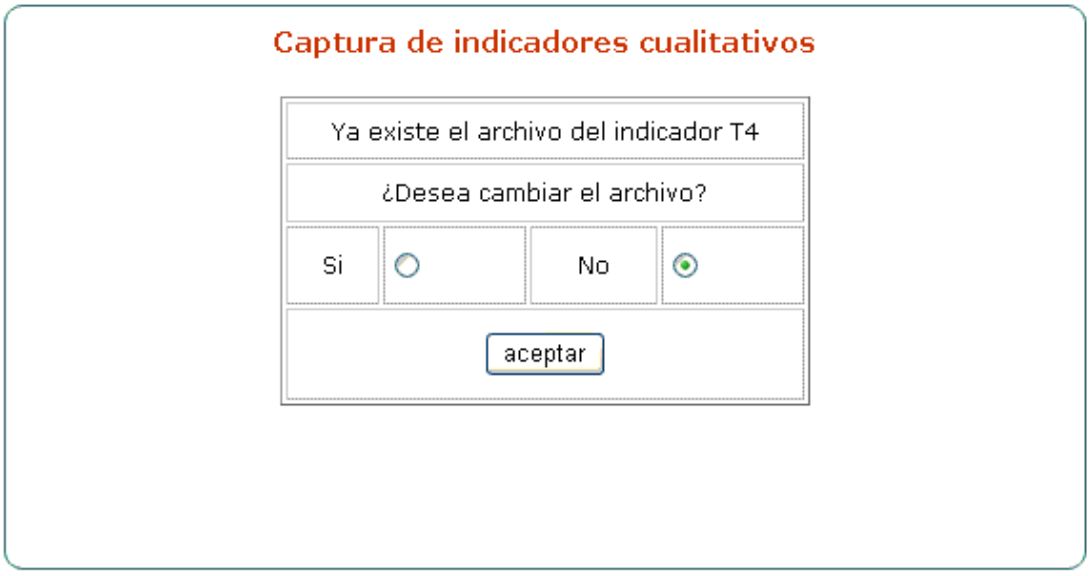
Pantalla A3.6



Pantalla A3.6.1

Si es necesario cambiar totalmente o el mayor número de registros en un indicador cualitativo, entonces podrá generarse como una nueva captura seleccionando el nombre del indicador y llenar nuevamente todos los campos.

Al presionar el botón **Guardar**, se le confirmará si es el caso, la existencia del archivo del indicador y le proporcionará la posibilidad de reemplazarlo o no. (ver Pantalla A3.7)



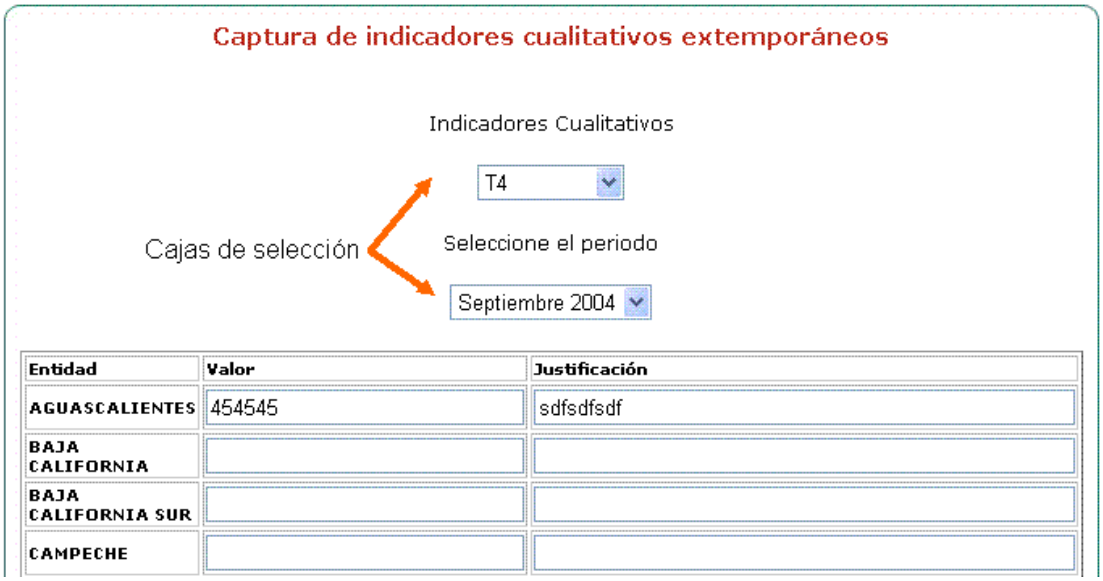
Pantalla A3.7

CAPTURA DE INDICADORES EXTEMPORANEOS CUALITATIVOS

Para crear un indicador cualitativo extemporáneo, necesitamos seleccionar el nombre del indicador y el periodo en su respectiva caja de selección. El periodo que se muestra en la pantalla corresponde a un mes fuera del periodo actual próximo a cargar.
Posteriormente ingrese valores y la justificación en cada estado de la lista mostrada y presione el botón **Guardar**. (ver Pantalla A3.8)

Las validaciones en esta pantalla:

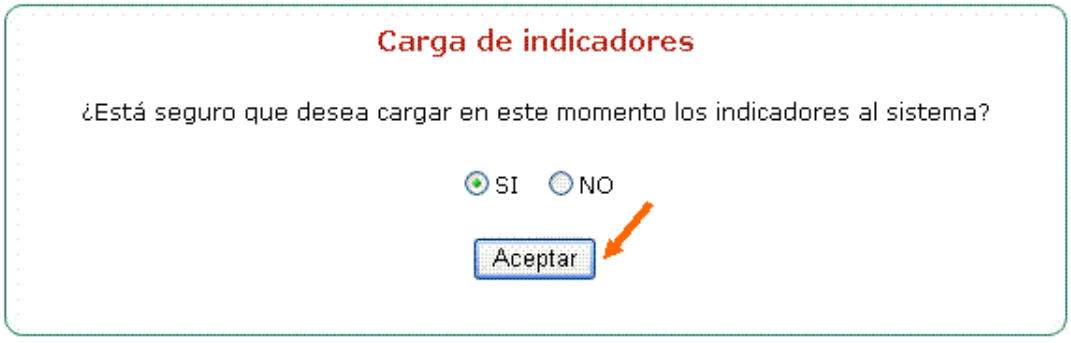
- Elegir de la caja de selección un indicador.
- Elegir de la caja de selección un periodo.
- El campo de texto valor únicamente es numérico.
- El campo de texto justificación es alfanumérico y no permite caracteres especiales.
- Todo valor deberá tener una justificación o viceversa.



Pantalla A3.8

Por último, **todos** los indicadores por archivo txt y captura guardados en un directorio del sistema podrán ser almacenados en la base de datos en el mes correspondiente al periodo que muestran las pantallas, en este caso Diciembre 2004.

Esta pantalla será desplegada desde la opción del menú Cargar Indicadores.
Es en esta función donde intervendrá la aplicación de escritorio ETL que lleva a la base de datos los valores correspondientes de cada archivo txt de los indicadores, así como las operaciones debidas con las variables. (ver Pantalla A3.9)



PantallaA3.9

Aplicación ETL.vb (Extraction, Transference and Load).

Extracción, transferencia y carga de los archivos txt a la base de datos, para cada uno de los 72 indicadores. La aplicación ETL es creada como una aplicación de escritorio. Las funciones principales del ETL sobre los archivos txt son:

- ✓ Abrir el archivo txt
- ✓ Realiza un conteo de 32 líneas en lista que representan a los estados de la república mexicana
- ✓ Valida el número de columnas que corresponden al número de variables asignadas según el indicador
- ✓ Valida el tipo de carácter numérico o alfanumérico
- ✓ Selecciona línea por línea el valor de las variables para realizar operaciones y obtener resultados
- ✓ Inserta a la base de datos de acuerdo al indicador, variables y resultados.

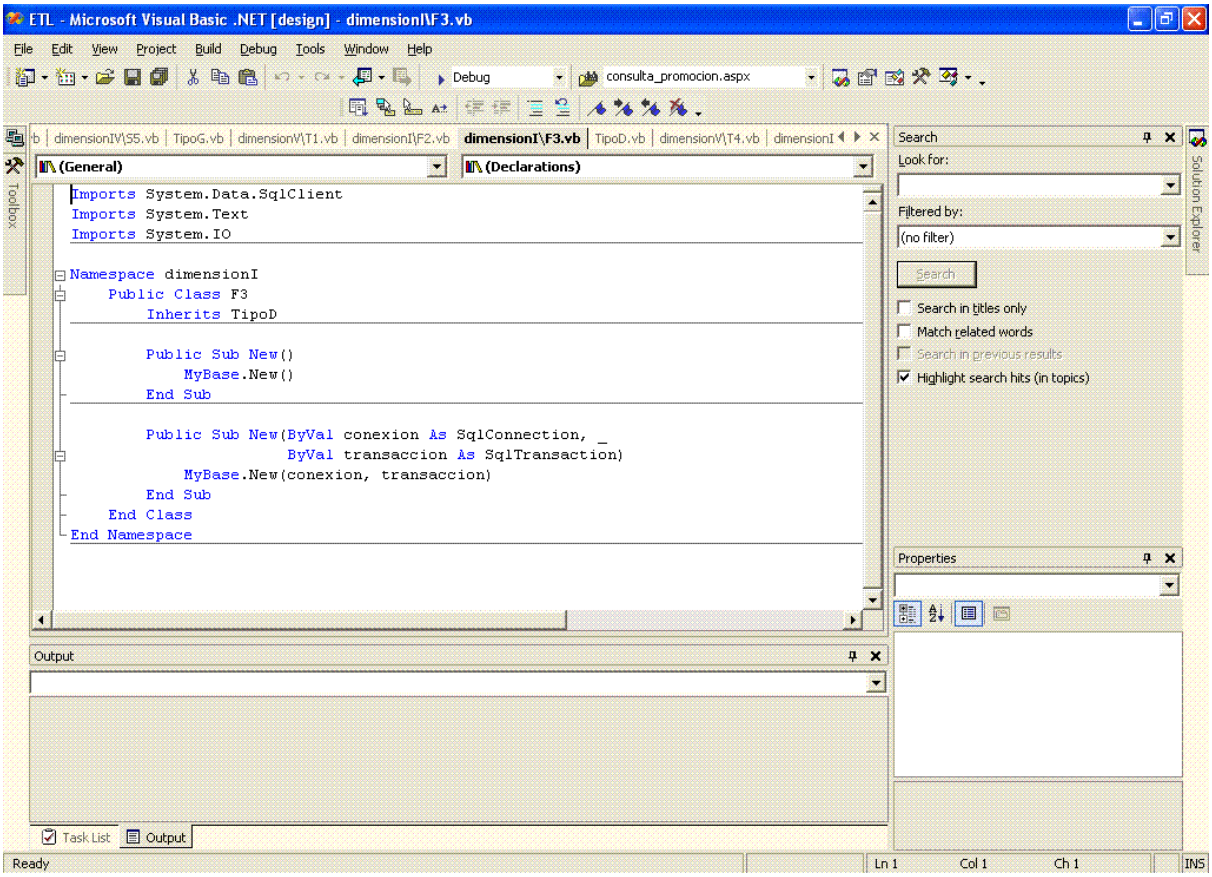
La siguiente pantalla muestra el editor de desarrollo en Visual Basic.Net para programar la clase del indicador F3.vb (ver Pantalla A3.10)

Ubicación de clases y librerías utilizadas en la clase F3:

Import System.Data.SqlClient
Import System.Text
Import System.IO

Namespace dimensionI:
El indicador F3 pertenece a la dimensionI y cada uno de los indicadores pertenece a una de las cinco dimensiones.
Public Class F3:
Inicio de la clase F3
Inherits TipoD: Herencia de la clase TipoD a la clase F3
Public Sub New() y Public Sub New(ByVal conexion As SqlConnection, ByVal transaccion As SqlTransaction):

Sobrecarga de constructores



Pantalla A3.10

Los indicadores se caracterizan por pertenecer a una dimensión, tener un número de variables, realizar una operación aritmética con sus variables y ajustarlas a una ponderación específica. Por ello, los indicadores también pertenecen a un TipoA, TipoB, TipoC, TipoD y TipoE, que identifica y reúne a los indicadores que tengan el mismo número de variables, con el fin de realizar una clase con la operación y reutilizar código heredando a las demás clases de indicadores que lo requieran.

La siguiente pantalla muestra el método CalcularFormula que realiza una operación con sus variables para el indicador S5.vb (ver Pantalla A3.11)

Namespace dimensionIV:

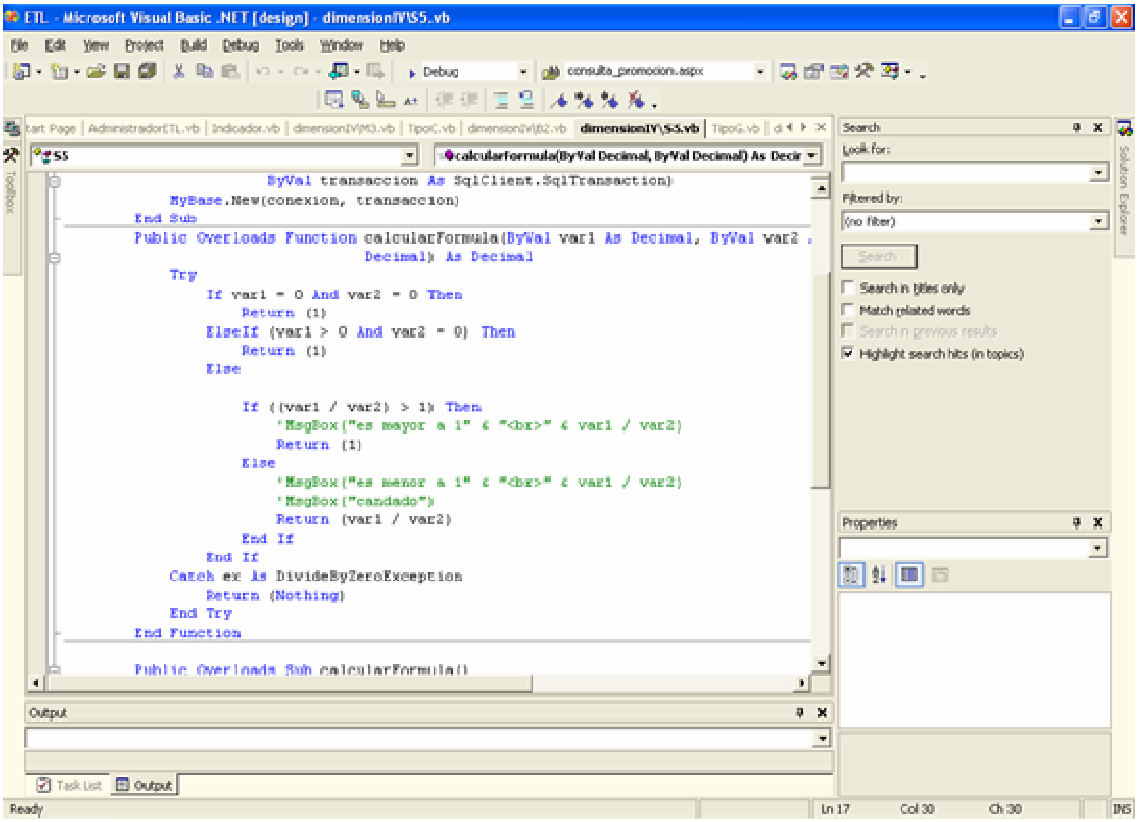
El indicador S5 pertenece a la dimensión IV

Public Overloads Function calcularFormula(ByVal var1 As Decimal, ByVal var2 As Decimal), As Decimal:

El overloads lo identifica como "Sobrecarga de método calcularFormula"

Try Catch ex As DivideByZeroException End Try:

Excepciones, si el divisor fuera cero entonces mandaría el mensaje de no resolver la operación por la existencia de un cero.

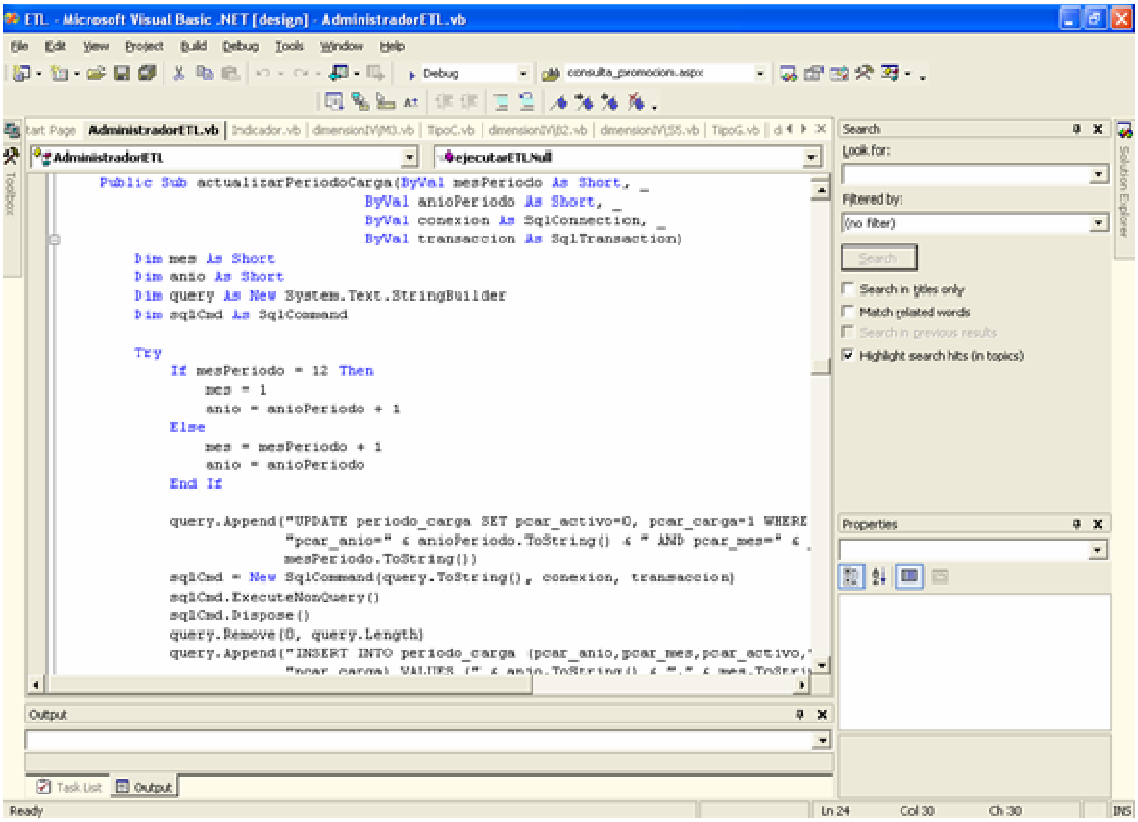


Pantalla A3.11

La clase AdministradorETL.vb: (ver Pantalla A3.12)

- Integra todos los nombres de los indicadores que serán reconocidos en el proyecto de la aplicación.
- Realiza las conexiones a la base de datos.

- ❖ Realiza operaciones de inserción a la base de datos para la carga de información de los indicadores con respecto al periodo en el que se deseen subir, es decir, si la carga es ordinaria o extemporánea.
- ❖ Actualiza consecutivamente los periodos próximos a cargar información.
- ❖ Realiza búsquedas en las tablas de los indicadores.
- ❖ El método EjecutarETL integra las funciones que mantienen lista la información de los indicadores en archivos txt, listas para ser cargadas a la base de datos.



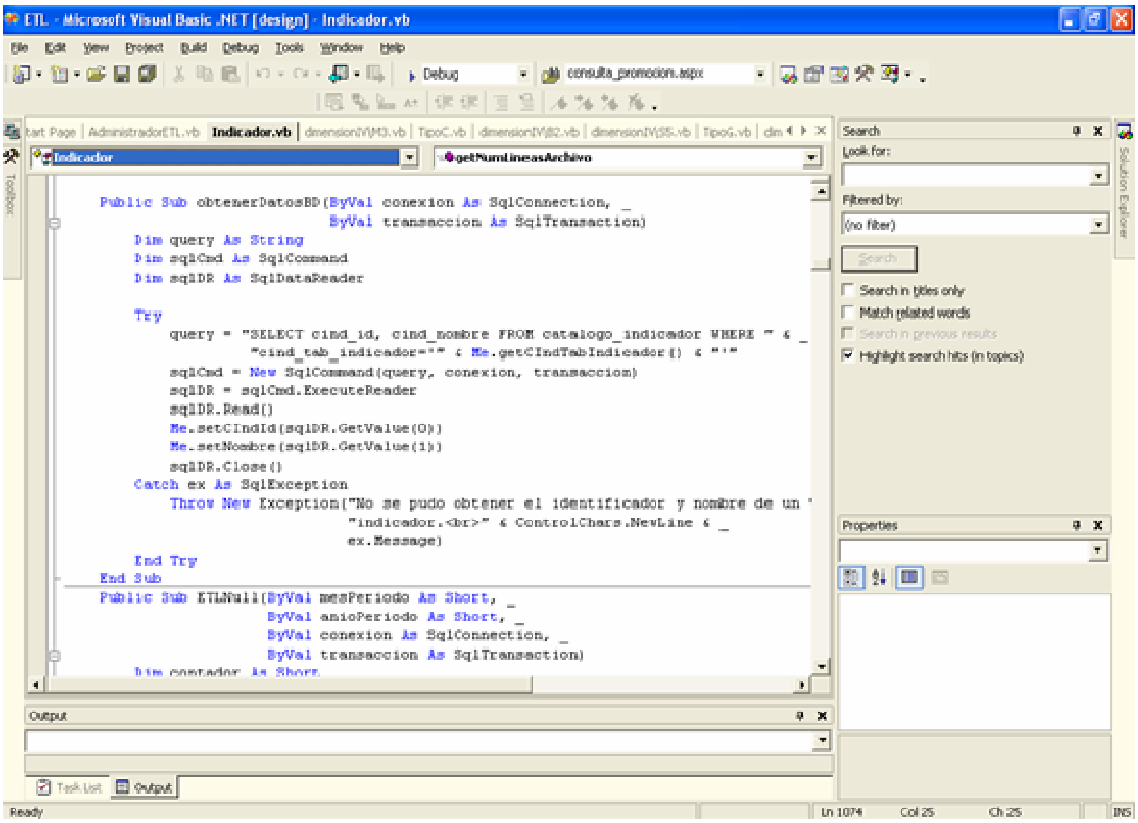
Pantalla A3.12

La clase Indicador.vb: (ver Pantalla A3.13)

- ❖ Contiene un método que permite abrir el archivo con el nombre del indicador, selecciona línea por línea asignándolos a variables en un ciclo de 32 registros por variable.
- ❖ Método que valida el formato de las variables, el número de columnas de variables.
- ❖ Método general que realizará las operaciones aritméticas de cada indicador.

- Realiza búsquedas en las tablas de los indicadores.
- Realiza inserciones de los valores extraídos del archivo txt a la tabla del indicador correspondiente.

- Actualiza valores extraídos del archivo txt a la tabla del indicador correspondiente.



Pantalla A3.13

El contenido del proyecto ETL reúne una clase por indicador (72 indicadores), una clase por dimensión (5 dimensiones), una clase AdministradorETL.vb e Indicador.vb que integra los métodos que ejecutaran los indicadores.

Pruebas

1. Ingreso y validación de información al subir la información de los indicadores por medio de archivos y en la captura de los indicadores cualitativos para crear sus archivos con extensión .txt
2. Ingreso y comportamiento de la base de datos de los archivos .txt

3. Instalación de la aplicación ETL y CargaInformación en diferentes equipos PC.

Implementación del sistema

El sistema actualmente se encuentra alojado en servidores de la DGSCA, la integración de todos los módulos involucró varias pruebas tanto en forma independiente como en conjunto para su correcta operación. Se cuenta con respaldos del sistema desarrollado y de la información que es almacenada a la base de datos considerando que primero es guardada en el disco duro del servidor que aloja el sistema.

Conclusiones

La actividad de desarrollar sistemas sobre web reconoce las habilidades para una buena administración en el tiempo, en la organización de documentos y códigos. Durante el desarrollo del sistema se crearon respaldos cada 15 o 30 días con el fin de llevar un control y avance del sistema, por su gran amplitud, tener como respaldo siempre un avance final de una parte y no volver a comenzar o perderlo todo.

Se desarrollaron dos aplicaciones para funcionar como una sola. La aplicación CargaInformación. aspx.vb para guardar la información por medio de archivos al disco duro del servidor desde una página en Internet y después de haber sido verificado el formato correcto por el sistema, entonces indicar en una de las opciones del menú de la página almacenar en la base de datos, actividad realizada por la aplicación ETL.vb que vuelve a verificar la información de los archivos.

El funcionamiento del sistema se realizó en los tiempos planeados y permitió que los demás módulos pudieran continuar su desarrollo con la información ya almacenada en la base de datos.

La prioridad de las actividades en un sistema de esta amplitud requiere gran responsabilidad en su análisis y programación principalmente para brindar resultados veraces y confiables.

- Instalación de sistema operativo Linux y servicios Apache, PHP y Mysql.
- Instalación de servicios Apache y Mysql en sistema operativo Windows XP.
- Asesorías a personal dentro de la Subdirección de Servicios Web de la DGSCA: prestadores de servicio social y becarios.

  Temas:

  - Diseño HTML y hojas de estilo Cascading Style Sheets (CSS)
  - Programación en PHP
  - Programación en Javascript
  - Programación Visual Basic.Net
  - Uso de manejadores de base de datos Mysql y SQL Server 2000
  - Uso de sentencias SQL
  - Uso de software ERWIN para diseño de bases de datos
  - Uso de sistema operativo Linux
  - Comandos para manipulación de archivos
  - Uso de SSH para conexión a servidores

Conclusiones generales

Se utilizaron las metodologías que cumplían las expectativas para el desarrollo de los sistemas, además las posibles combinaciones para el logro y realización de cada una de las actividades. El análisis, diseño e implementación de cada actividad se realizó en conjunto con el equipo profesional de la DGSCA. El objetivo principal en las tres actividades fue ofrecer información oportuna y confiable.

Para el desarrollo de los sistemas fue necesario un período de análisis para resolver el problema con el fin de detectar y dar solución a cada una de las necesidades de información general y particular de personas involucradas. Los sistemas desarrollados cumplieron con los requerimientos del usuario.

En cada una de las actividades se demostró el enorme enlace que tiene la computación y los mecanismos para administrar la información, y ahora las ventajas proporcionadas por estos sistemas permite almacenarla virtualmente y presentarla en el momento que los responsables de esta la requieran.

Uno de los aspectos más importantes aprendidos durante las primeras etapas de análisis fue la comunicación con el usuario, reconociendo que la necesidad de éste debe entenderse en un lenguaje natural y no técnico.

Se considero el ambiente gráfico, con la finalidad de que el sistema fuera más amigable y de fácil navegación.

Para el desarrollo de los sistemas se utilizaron:

- Lenguajes de programación para sistemas Web "PHP 5, Visual Basic .NET, en su versión Microsoft Visual Studio.Net 2003 Framework 1.1 y Javascript".
- La aplicación de la base de datos, en este caso, SQL Server 2000 y MySQL permitió el acceso multiusuario para diversas vistas al mismo tiempo y aprovechar el sistema de red.
- El lenguaje estándar de SQL para bases de datos permitió facilidad en la realización de consultas, altas y bajas: "SQL Server 2000" utilizado en el sistema "Sistema de Evaluación del Desempeño de los Servicios Estatales del Empleo" y "MySQL", utilizado en los sistemas correspondientes al sistema de "Diplomado Integral de Telecomunicaciones" y "Red de

Macrouniversidades para América Latina y el Caribe", por ello, la elección de estas para aprovechar todas las características y precauciones que debe tener el enfoque relacional.

- Visual Basic.NET, lenguaje de programación con editor de entorno de desarrollo gráfico para aplicaciones de escritorio y web, permitió aprovechar las clases requeridas para realizar las operaciones solicitadas por el cliente.

Se aprovecharon los equipos tipo PC para desarrollar los sistemas y adaptarse a las necesidades del usuario considerando los *recursos* que existen.

Con la programación en PHP y Visual Basic.NET, el código es sencillo de depurar, un primer paso es realizar un análisis de las variables utilizadas dentro del proyecto, para continuar con la interpretación del código, lo que evita el proceso de compilado y ligado de los compiladores convencionales.

Se evaluaron los requerimientos de hardware para que el sistema tuviera un tiempo de respuesta aceptable, además se utilizaron los equipos que se encontraron disponibles dentro de la SSW de la DGSCA, evaluando con estándares establecidos por Microsoft y Software Libre.

El modelo de datos es uno de los puntos más importantes el cual pretende una visión coherente de la información que se mantiene para cada uno de los casos, permitiendo la fácil interpretación, visión que deberá identificar los diferentes tipos de entidades, atributos y relaciones, de tal manera que la información esté completamente normalizada.

Es importante que en el desarrollo de un sistema con bases de datos, mediante el uso de una metodología apropiada, se conjunten datos, procesos y tecnologías para obtener soluciones adecuadas a las necesidades.

En síntesis las características y ventajas que se obtuvieron desarrollando estos sistemas fueron:

a) Logro de objetivos deseados en la **planeación.**
b) **Aprovechamiento** máximo de los recursos de cómputo.

c) La **integración** de toda la información.

d) Contar con información **oportuna y confiable.**

e) La **delimitación de responsabilidades** dando lugar a la toma de decisiones adecuada y oportuna.

f) El **desempeño eficiente** para satisfacer requerimientos generales y particulares.

El impacto al desarrollar sistemas web a usuarios de la universidad UNAM y de gobierno son soluciones reales que implican una gran responsabilidad, el manejo de la información oportuna y confiable, facilidad de operación en el sistema y automatizar en lo posible las tareas, con el fin de evitar que el usuario realice cálculos largos y vulnerables a cometer errores.

Estas actividades me permitieron ver la solidez de la formación profesional como Ingeniera en Computación y confrontar mis conocimientos para enfrentar problemáticas en forma real, siendo de gran importancia para el futuro y en lo posterior dar solución a problemas dentro de las organizaciones a través de sistemas de cómputo con ética profesional y conocimiento de las metodologías existentes.

La carrera de Ingeniería en Computación me ha brindado la oportunidad de contar con mayores perspectivas de desarrollo, por el alto nivel intelectual y tecnológico de las ciencias que utiliza.

Después de la formación profesional, dependerá de aquí en delante de los preceptos de innovación, actualización, creatividad y capacidad de poder resolver problemas, con lo aprendido durante los estudios en esta área.

Anaid Victoria Velázquez Rivera

BIBLIOGRAFIA

1. Cruz Heras, Daniel, Flash, PHP y MySQL: contenidos dinámicos, Madrid: Anaya Multimedia, c2006.
2. Gilmore, W. Jason, Beginning PHP an MySQL 5: from novice to profesional : Berkeley, California: Apress, C2006.
3. Welling, Luke, PHP and MySQL Web development / Luke Welling and Laura Indianapolis, Indiana: Sams, c2005
4. Gallego Vazquez, Jose Antonio, Desarrollo Web con PHP y MySQL, Madrid: Anaya Multimedia,c2003.
5. Richard E. Fairley , Ingeniería de Softwarel: Mc Graw-Hill
6. Pressman Roger, Ingeniería de Software Prentice-Hall
7. Edward Yourdon , Análisis Estructurado Modernol: Prentice-Hall
8. Kenneth E. Kendall and Julie E. Kendall, Análisis y Diseño de Sistemas: Prentice-Hall Hispanoamericana, S.A.
9. F. Korth Henry, Silberschatz Abraham, Fundamentos de Bases de Datos: Mc Graw-Hill
10. Carlo Batini, Stefano Ceri, Shamkant B. Navathe, Diseño Conceptual de Bases de Datos: Addison-Wesley / Díaz de Santos
11. James Martín, Organización de las bases de datos: Prentice-Hall Hispanoamericana
12. Larousse, Gran Enciclopedia Larousse
13. Héctor G. Tejera, Diccionario Moderno de Informática: Grupo Editorial Iberoamericana S.A. de C.V.