

Universidad de las Ciencias Informáticas
Facultad Regional “Mártires de Artemisa”



*TÍTULO: Base de datos para la Dirección de Educación
de la Administración Provincial de Artemisa.*

*Trabajo de Diploma para optar por el Título de Ingeniero
en Ciencias Informáticas*

AUTORA: Odélaisy Badia Casola.

TUTOR; Ing. Anaibis Álvarez Morales.

CO-TUTOR; Lic. José Ángel Dieppa.

Artemisa, Cuba.

Junio, 2012



"Seamos realistas y hagamos lo imposible."

Ernesto Che Guevara

Dedicatoria

*A mi familia, que es mi razón de ser.
Especialmente a mis padres y abuelos aunque uno ya no esté.
A mis hermanos y amigos que son mi luz.*

Agradecimientos

A mis padres por enseñarme todo lo que se, por hacer posible que llegara tan lejos en mi vida, por su sacrificio y dedicación. Por ser el mejor ejemplo que toda hija pueda tener, por su comprensión, por saber guiarme siempre por el camino correcto, por confiar siempre en mí, por hacer posible que me convirtiera en la persona que soy ahora y por ser como son conmigo.

A mi familia por darme siempre su apoyo incondicional, su cariño y tantas alegrías juntas. A mis hermanos que siempre están ahí cuando los necesito, a una persona que considero mi hermana aunque no lo sea ya que siempre ha estado ahí en las buenas y malas y ella sabe quién es.

A mis tres abuelos que me quedan y en especial al que ya no está por guiarme siempre a ser mejor mujer y enfrentar cualquier problema, a mi tío que lo considero como mi papa.

A mi tutora Anaibis Álvarez y mi cotutor José Ángel Dieppa que se han convertido en parte de mi familia y más que eso por confiar en mí.

A los profesores Humberto Santos y Ulises Socas porque sin su apoyo no estaría hoy donde estoy.

A todos mis amigos, y no menciono nombres por si se me queda alguno, no se me pongan celosos.

Muchas gracias a todos por confiar en mí.....

Odelaisy.

Declaración de Auditoría

Declaro que soy el único autor de este trabajo y autorizo al Centro de Desarrollo de la Facultad Regional “Mártires de Artemisa” de la Universidad de las Ciencias Informáticas (UCI) a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de junio del año 2012.

Odelaisy Badia Casola

Lic. José Ángel Dieppa.

Ing. Anaibis Alvarez Morales.

Datos de Contacto

Tutor: Ing. Anaibis Alvarez Morales.

Correo Electrónico: aamorales@hab.uci.cu

Co-Tutor: Lic. José Ángel Dieppa.

Correo Electrónico: jadb@hab.uci.cu

Resumen

Con la división político administrativa adoptada por el país en el año 2011 surge la provincia de Artemisa, donde se crea la Administración Provincial (AP) la cual se encuentra situada en el municipio cabecera. La Administración Provincial cuenta con 32 Direcciones que manejan grandes volúmenes de información de forma manual, por lo que la misma se encuentra en fuentes no seguras. En aras de mitigar o erradicar dichos problemas, se hace necesaria la creación de un sistema de gestión de la información para la Administración Provincial (AP). Con este fin el Centro de desarrollo de la Facultad Regional Mártires de Artemisa de la Universidad de las Ciencias Informáticas (UCI), da inicio al proyecto Sistema Informativo de la Administración Provincial de Artemisa (SINAP), el cual está dividido en varias direcciones y departamentos, entre los que se encuentra el Módulo de la Dirección de Educación.

Esta dirección necesita de una base dato con su respectiva capa de acceso a datos para lograr una mejor integridad, confiabilidad y centralidad de de la información en la Dirección de Educación de la Administración Provincial de Artemisa. Para dar cumplimiento a la necesidad antes expuesta, la metodología de desarrollo de software utilizada fue la SXP, la cual permitió agilizar el análisis, diseño e implementación de la solución propuesta. La arquitectura que se seleccionó fue la Arquitectura N-Capas. Asimismo se validó la integridad de los datos, la calidad, y el alto rendimiento de la solución propuesta.

De esta manera se obtuvo como resultado la base de datos con su respectiva capa de acceso a datos en su versión 1.0 elevando así los niveles de integridad, confiabilidad y centralidad de la información.

Palabras Claves: base de datos, capa de acceso a datos, sistema gestor de base de datos.

Índice

<i>Introducción</i>	9
<i>Capítulo 1: Fundamentación Teórica</i>	8
1.1 Definición de términos informáticos utilizados	8
1.2 Sistemas de Gestión de Información. Tendencias Actuales	11
1.3 Metodologías de desarrollo de <i>software</i>	13
1.4 Introducción a las Bases de Datos	16
1.5 Herramientas y Tecnologías utilizadas.....	23
1.5.1 Sistema Gestor de Bases de Datos (SGBD)	23
1.5.2 Frameworks	26
1.5.3 Herramientas CASE (<i>Computer Aided Software Engineering</i>).....	27
1.5.4 Netbeans 7.01	29
1.5.5 Herramientas de control de versiones	30
<i>Capítulo 2: Diseño y Arquitectura de la base de datos</i>	32
2.1 Metodología para el diseño de bases de datos.	32
2.2 Técnicas para diseñar bases de datos relacionales	32
2.2.1 Normalización	33
2.3 Requisitos Funcionales y no Funcionales del Módulo de la Dirección de Educación.....	35
2.4 Diseño e implementación de la Capa de Acceso a Datos.....	37
2.4.1 Patrones de diseño utilizados.....	37
2.4.2 Situación actual de la Dirección de Educación	39
2.4.3 Propuesta de Solución	40
2.4.4 Técnicas para crear Capas de Acceso a Datos.	40
2.5 Modelos empleados para la elaboración de la BD de la Dirección de Educación.....	41
2.6 Diagrama de Paquetes	45
2.7 Diagrama de Componente	45
2.8 Diagrama de despliegue:	48
2.9 Arquitectura de software	49
2.10 Implementación de la capa de Acceso	50
<i>Capítulo 3. Adquisición y validación de los resultados del sistema. ..</i>	56
3.1 Duplicidad e integridad de los datos	56
Integridad de los datos	56
3.2 Ejecución de pruebas	56
3.2.1 Pruebas de Conexión.....	57
3.2.2 Pruebas funcionales realizadas y resultados	58
3.3 Resultados y Funcionalidades obtenidas	61
3.4 Aporte Social y Económico	61

<i>Conclusiones Generales</i>	63
<i>Recomendaciones</i>	64
<i>Referencias Bibliográficas</i>	65
<i>Bibliografía</i>	68
<i>Anexos</i>	69
<i>Glosario</i>	72

Índice de Tablas y Figuras

<i>Tabla 1 Requisitos Funcionales</i>	35
<i>Tabla 2 Descripción de la entidad del modelo</i>	43
<i>Tabla 3 Descripción de la entidad Aprendizaje del alumno</i>	43
<i>Tabla 4 Descripción de la entidad Matrícula EIDE y EVA</i>	44
<i>Tabla 5 Descripción de la entidad modalidades</i>	44
<i>Figura 1 Árbol invertido</i>	21
<i>Figura 2 Diagrama Entidad-Relación</i>	42
<i>Figura 3 Diagrama de paquetes para la Dirección de Educación</i>	45
<i>Figura 4 Diagrama de Componente</i>	46
<i>Figura 5 Diagrama de Despliegue</i>	49
<i>Figura 6 Insertar datos generales</i>	60
<i>Figura 7 Modificar datos generales</i>	60
<i>Figura 8 Eliminar datos generales</i>	61

Introducción

Introducción

La información siempre ha sido un recurso muypreciado. Desde la antigüedad los datos recopilados se almacenaban en bibliotecas u otros lugares donde estarían seguros. El objetivo de ese proceso era poder reutilizar la información cuando fuere necesario.

Hoy se vive la tercera gran revolución tecnológica. Las computadoras, conectadas a través de la Internet, los medios de comunicación interactiva, la realidad virtual y otros avances en el área de la informática, transformaron radicalmente las nociones del tiempo y del espacio e incluso de la realidad.

El hombre ha logrado explorar su medio físico y su propia realidad como ser vivo. Especialistas, empresas, universidades y gobiernos trabajan de conjunto e inician, cada vez más, ambiciosos proyectos con tecnologías de punta. Aumenta la competitividad en los mercados. La ciencia se ha transformado en un instrumento de poder económico que incide en los distintos ámbitos de la vida humana.

Los sistemas de información administrativa han ganado un protagonismo indiscutible al facilitar la planificación y toma de decisiones en lugares inimaginables. A medida que se hacen más rápidos y confiables, mayor serán las posibilidades de los dirigentes administrativos de las empresas de tomar las decisiones correctas en el momento adecuado. Las entidades estatales son lugares donde se puede explotar al máximo las posibilidades de estos sistemas de gestión administrativa siempre que los responsables sepan usarlos al máximo.

La evolución de los sistemas de información ha tenido una repercusión considerable en la gestión de los datos. En la década de los sesenta aparecieron las computadoras personales y con ellas las grandes posibilidades para automatizar tareas y procesos intrínsecos de las organizaciones empresariales. Con estos avances se logró desplazar el centro de gravedad de la informática, que hasta aquel momento eran los procesos, hacia la estructuración de los datos.

Introducción

A finales de los años sesenta y principios de los setenta la gestión de los datos había dado un gran salto conocido como “la primera generación de base de datos”. Poco tiempo después Edgar *Codd* planteó su modelo relacional, considerado como “la segunda generación de bases de datos”, lo que impactaría directamente en la implementación para productos comerciales posteriormente.

En los últimos años se ha sido testigo de un avance sin precedentes de la tecnología de las base de datos a través de multimedias activas, deductivas, orientadas a objetos, seguras, temporales y móviles. Surgen nuevas aplicaciones de ingeniería asistida por computadoras, sistemas de información geográfica, aplicaciones científicas, sistemas de comercio electrónico, aplicaciones para la educación, la formación y la salud.

El Sistema de Gestión de Bases de Datos (SGBD) reporta una serie de ventajas en el proceso de manipulación y gestión de la información como: el control sobre la redundancia, la compartición y el suministro de datos para otras aplicaciones o sistemas. Las empresas que emplean estas Tecnologías de la Información y las Comunicaciones elevan sus índices de productividad y eficacia.

Desde el triunfo revolucionario de 1959 se propuso potenciar la ciencia y las tecnologías a todos los niveles posibles. En Cuba el Ministerio de la Informática y las Comunicaciones (MIC) regula las telecomunicaciones, la informática, las redes, los servicios de valor agregado de las info-comunicaciones, la automatización, la industria electrónica... Garantiza sobre bases democráticas y equitativas la estrategia de desarrollar la Industria Cubana de las Aplicaciones Informáticas.

Siguiendo esta la política del estado cubano de desarrollar la industria del *software* se crea en el año 2002 la Universidad de las Ciencias Informáticas (UCI). Posee profesionales con una alta preparación científica y política. Al mismo tiempo contribuye al desarrollo de la industria del *software* cubano. Para hacer extensivo este desarrollo de las ciencias informáticas hacia todas las regiones de Cuba, en el año 2007 se crean tres Facultades Regionales de la UCI en las provincias de

Introducción

Granma, Ciego de Ávila y en el municipio de Artemisa que en aquel entonces pertenecía a la provincia de La Habana.

Con la división político administrativa adoptada por el país en el año 2011 surge la provincia de Artemisa. La Administración Provincial se encuentra situada en el municipio cabecera esta cuenta con 32 Direcciones entre las cuales se encuentra la Dirección de Educación donde se tramita toda la información relacionada con el seguimiento, control y fomento de la actividad educacional en la provincia. Actualmente existen en la Dirección de Educación un conjunto de dificultades en cuanto al manejo de la información pues los datos se encuentran en fuentes no seguras: en formato duro o de manera digital en libros de cálculo de Microsoft Excel. Esta manera de proceder con los datos genera descontrol y desorganización; se compromete la integridad y confiabilidad de los datos, ocurren modificaciones no deseadas y se descentraliza la información.

En la dinámica de trabajo de la Dirección de Educación se invierte gran cantidad de tiempo y fuerza de trabajo en la búsqueda y consulta de los datos que se manejan como por ejemplo las altas y bajas estudiantiles, la disponibilidad de profesionales, matrículas de estudiantes, entrega de materiales docentes y las proyecciones de cada curso escolar por escuela, en ocasiones la información no se encuentra disponible, resultando complicado mantener informados a todos los trabajadores de los diferentes departamentos que conforma la Dirección.

A partir de la problemática descrita se plantea el siguiente **problema de investigación**: ¿Cómo contribuir a la integridad, confiabilidad y centralidad de la información, en la Dirección de Educación del Sistema Informativo de la Administración Provincial de Artemisa?

Objeto de estudio: procesos de gestión de información en los sistemas gubernamentales.

Campo de acción: base de datos relativa a los procesos de gestión de la información en la Educación.

Introducción

Teniendo en cuenta el problema expuesto anteriormente se traza el siguiente **objetivo general**: Desarrollar una base de datos que contribuya a la integridad, confiabilidad y centralidad de la información, en la Dirección de Educación de la Administración Provincial de Artemisa.

Objetivos específicos:

1. Establecer los fundamentos teóricos metodológicos relacionados con el desarrollo de base de datos.
2. Desarrollar la base de datos de acuerdo a las características de integridad, confiabilidad y centralidad de la información en la Dirección de Educación de la Administración Provincial de Artemisa.
3. Validar la base de datos que contribuya a la integridad, confiabilidad y centralidad de la información en la Dirección de Educación de la Administración Provincial de Artemisa.

En la investigación se formula la siguiente **idea a defender**: Con el desarrollo de la base de datos se contribuirá a la integridad, confiabilidad y centralidad de la información en la Dirección de Educación de la Administración Provincial de Artemisa.

En la presente investigación se emplearon los siguientes Métodos teóricos:

Histórico-lógico: permitió consultar y estudiar la bibliografía existente sobre los sistemas de Bases de Datos, para realizar un estudio histórico de estos sistemas.

Analítico-sintético: para el procesamiento de la información y arribar a las conclusiones de la investigación, así como precisar las características del modelo de datos que se propone diseñar.

Introducción

Inductivo-deductivo: a partir del planteamiento de la idea a defender y siguiendo la lógica de deducción se llega a nuevos conocimientos y predicciones que son sometidos a verificaciones. Se deduce que con el desarrollo de la base de datos para la Dirección de Educación se perfeccionará la integridad, confiabilidad y centralidad de la información en la Dirección de Educación de la Administración Provincial de Artemisa.

Modelación: Este método permitió modelar el proceso de diseño e implementación del sistema a través de la creación de modelos que representan abstracciones, con el objetivo de explicar cómo funcionan los procesos.

En la presente investigación se emplearon los siguientes Métodos empíricos:

Análisis documental: El uso de este método resultó útil para analizar toda la documentación existente en la Dirección de Educación.

Población y Muestra: La población definida en la investigación está constituida por 30 trabajadores de la Dirección de Educación de la de la Administración Provincial de Artemisa con la selección de una muestra de 10 personas representando el 33 % de la población; la técnica seleccionada fue el muestreo intencional.

Variables Independientes: integridad, confiabilidad y centralidad de los datos.

Variable Dependiente: Base de Datos para la Dirección de Educación de la Administración Provincial de Artemisa.

Aporte Práctico:

El desarrollo de la base de datos para la Dirección de Educación de la Administración Provincial de Artemisa constituye una alternativa sin precedentes en los gobiernos provinciales de Cuba. Complementa la estrategia de gestión de la información del proyecto “Sistema Informativo de la Administración Provincial”

Introducción

(SINAP), permitiendo la integración cliente-servidor-base de datos del sistema en desarrollo.

La mayoría de los sistemas informáticos modernos que trabajan con un Sistema Gestor de Bases de Datos, requieren una Capa de Acceso a Datos o Capa de Persistencia, la cual es un componente fundamental en los sistemas informáticos, principalmente los sistemas de tipo aplicación web. La capa de acceso a datos es una porción de código que se encarga de realizar el acceso a los datos. Su origen se encuentra vinculado con el patrón Modelo-Vista- Controlador.

Actualmente este patrón es muy utilizado en las aplicaciones que necesiten vincularse con bases de datos.

Los tres capítulos están estructurados de la siguiente manera:

En el Capítulo 1. Fundamentación teórica: Se hace alusión al estado del arte del tema tratado, las tendencias, técnicas, tecnologías, metodologías y *software* usados para la solución del problema.

En el Capítulo 2. Diseño y Arquitectura de la base de datos: En este capítulo se le da seguimiento al desarrollo de la base de datos. Marcando cada paso con una serie de artefactos y documentos.

En el Capítulo 3. Adquisición y validación de los resultados del sistema: Se verá lo referente a la validación teórica del diseño, la integridad y el análisis de la redundancia. También se describirán los tipos de pruebas identificados y se explicara la prueba realizada a la BD.

Capítulo 1: Fundamentación Teórica.

Capítulo 1: Fundamentación Teórica

Introducción

Con el desarrollo de este capítulo se presentan un grupo de conceptos a los que se hacen referencia en el resto del trabajo, se exponen los temas relacionados con el campo de acción para su mejor comprensión. Se abordan aspectos relacionados con las bases de datos, tales como su definición, arquitectura y tipos considerando que actualmente el uso de las bases de datos es fundamental en diferentes esferas de la vida social. Todo este análisis se realiza teniendo en cuenta las necesidades y características del medio donde se aplicará la solución propuesta.

1.1 Definición de términos informáticos utilizados

Gestión

Gestión del latín gestio: acción de administrar. Es la actividad profesional destinada a establecer los objetivos y medios para la realización, organización del sistema, elaboración de la estrategia de desarrollo y ejecución de la gestión del personal. El término gestión, por lo tanto, implica al conjunto de trámites que se llevan a cabo para resolver un asunto o concretar un proyecto, se refiere al conjunto de actividades que desarrollan, movilizan y motivan al personal empleado que una empresa necesita para su éxito. **(1)**

El concepto de gestión, tal como se le utiliza actualmente, proviene del mundo de la empresa y atañe a la gerencia. La gestión se define como la ejecución y el monitoreo de los mecanismos, las acciones y las medidas necesarias para el cumplimiento de los objetivos de la institución. La gestión, por consiguiente, implica un fuerte compromiso de sus actores con la institución y también con los valores y principios de eficacia y eficiencia de las acciones ejecutadas. **(2)**

De los planteamientos antes mencionados sobre lo que significa la palabra gestión, la referencia que más se relaciona es el segundo concepto, por tal motivo será usado en todo el proceso de desarrollo.

Capítulo 1: Fundamentación Teórica.

Información

Múltiples trabajos se han dedicado a indagar sobre el término información y su importancia como recurso indispensable para la sociedad. El concepto de información es entendido de diferentes maneras y con diferentes significados en todos los países del mundo.

La información es un conjunto organizado de datos procesados, que constituye un mensaje sobre un cierto fenómeno. La información permite resolver problemas y tomar decisiones, ya que su uso racional es la base del conocimiento. Esta es un fenómeno que aporta significado o sentido a las cosas, ya que mediante códigos y conjuntos de datos, forma los modelos de pensamiento humano. En términos generales, se habla de la información como un conjunto de datos que están organizados y que tienen un significado. De esta manera, si se toman los datos por separado no tendrían un significado mientras que si se agrupan en forma organizada, sí. La información es un elemento fundamental en el proceso de la comunicación, porque tiene un significado para quien la recibe, que la va a comprender si comparte el mismo código de quien la envía. **(3)**

La información es un conjunto de datos con un significado, o sea, que reduce la incertidumbre o que aumenta el conocimiento de algo. En verdad, la información es un mensaje con significado en un determinado contexto, disponible para uso inmediato y que proporciona orientación a las acciones por el hecho de reducir el margen de incertidumbre con respecto a cada decisión. **(4)**

Gestión de la información

La gestión de la información es el proceso que se encarga de suministrar los recursos necesarios para la toma de decisiones, así como para mejorar los procesos, productos y servicios de la organización. La gestión de la información no es más que el proceso de analizar y utilizar la información que se ha conseguido y registrado para permitir tomar decisiones documentadas. La información para la

Capítulo 1: Fundamentación Teórica.

gestión es la información necesaria para tomar decisiones de gestión. Por lo tanto, la gestión de la información implica:

1. Determinar la información que se precisa.
2. Recoger y analizar la información.
3. Registrarla y recuperarla cuando sea necesaria.
4. Utilizarla.

Recuperación de la información

Proceso donde se accede a una información previamente almacenada, mediante herramientas informáticas que permiten establecer ecuaciones de búsqueda específicas. Dicha información ha debido de ser estructura previamente a su almacenamiento. El proceso de recuperación se lleva a cabo mediante consultas a la base de datos donde se almacena la información estructurada, mediante un lenguaje de interrogación adecuado. **(5)**

Capa de acceso a datos: Su función principal es almacenar y devolver datos a la capa de negocio, en algunos casos, que tengan procedimientos almacenados y funciones dentro de la capa. Es una arquitectura de tres capas y esta es la única que puede acceder a los mismos.

Integridad de la Información: es una característica que se busca en toda base de datos a fin de garantizar que una entidad (la cual puede ser una fila o un registro) siempre esté asociada a otras entidades que figuren en esa misma base. La integridad supone que todos los datos sean correctos, que no estén repetidos los registros, que no haya datos perdidos ni relaciones mal resueltas. Mencionar que describe la corrección de todos los elementos que presenta una base. Cuando se utilizan sentencias como INSERT, DELETE o UPDATE, la integridad de los datos puede verse afectada. Por ejemplo, se pueden añadir datos no válidos o

Capítulo 1: Fundamentación Teórica.

modificarse datos existentes en forma incorrecta, con lo que la integridad no se cumple. (6)

Confiabilidad de la Información: como la capacidad de un producto de realizar su función de la manera prevista. De otra forma, la confiabilidad se puede definir también como la probabilidad en que un producto realizará su función prevista sin incidentes por un período de tiempo especificado y bajo condiciones indicadas. Además, la ejecución de un análisis de la confiabilidad en un producto o un sistema debe incluir muchos tipos de exámenes para determinar cuan confiable es el producto o sistema que pretende analizarse. (7)

Centralidad de la Información: la centralización de los sistemas de información tiene la cualidad de permitir un control más sencillo, ya que es la mejor forma de captar, manipular y usar la información cuando es necesario que un gran número de usuarios puedan acceder a ella. Otra de sus ventajas es que evita la inconsistencia de las aplicaciones y de los programas de los departamentos. (8)

1.2 Sistemas de Gestión de Información. Tendencias Actuales

Debido al gran avance de las tecnologías de la informática y las comunicaciones, en el mundo existe una numerosa cantidad de sistemas de gestión de información, entre ellos los sistemas de gestión vinculados al sector educacional, la mayoría vinculados con universidades, pues ésta es la institución con más nivel tanto tecnológico como intelectual e inmersa en la búsqueda de soluciones a sus necesidades. Generalmente estos sistemas de gestión implantados en las universidades, responden solamente a los intereses trazados en dichos centros, por lo que resultaría muy difícil adaptar uno de estos sistemas a otro centro de enseñanza, siendo ello una de las causas fundamentales por la cual se tengan que desarrollar los sistemas de gestión de información con su respectiva capa de acceso a dato de acuerdo a cada institución.

Capítulo 1: Fundamentación Teórica.

Análisis de solución existente en el mundo

En el mundo, existen diversos sistemas de gestión de información que se han destacado por su eficiencia y facilidad de uso a la hora de interactuar con ellos.

UNESCO

Organizaciones como la UNESCO ha puesto en práctica sistemas que le han permitido una mejor gestión de la información en el sector de la Educación y cada uno cuenta con sus respectivas capas de acceso a datos, donde en diferentes países como la India, Tanzania y Senegal a puesto en práctica estos sistemas.

SIGA: Sistema Integrado de Gestión Académica

SIGA es un sistema de gestión desarrollado en España. Es una aplicación que puede ser utilizada en muchos centros educacionales, no solo universitarios, sino también en conservatorios, academias, colegios, centros de formación de empresas, maestrías, postgrados, entre otros.

Análisis de solución existente en Cuba

El acelerado avance en los conocimientos científicos y sus aplicaciones tecnológicas en el mundo actual, constituyen la base fundamental para el desarrollo económico y social de los pueblos. Cuba ha mantenido esta revolución científico-técnica, estableciendo el concepto de Informatización de la Sociedad en un entorno local, por lo que se ha propuesto la idea de digitalizar el quehacer de las direcciones del Gobierno, empezando desde sus municipios, mediante la creación de Sistemas para la Gestión de Información.

En la actualidad la Dirección de Ciencia del Ministerio de Educación Superior en Colaboración con la Facultad de Matemática y Computación de la Universidad de la Habana ejecutan un proyecto de desarrollo de sistemas de información para el control de la actividad científica y este también cuenta con su respectiva capa de acceso a datos.

El Instituto Superior Politécnico “José Antonio Echeverría ” (CUJAE) ha creado un sistema de gestión académica (SDI), desarrollado por el Grupo de Gestión

Capítulo 1: Fundamentación Teórica.

Universitaria Digital dirigido a aumentar la eficiencia de la gestión universitaria, específicamente el control de los procesos docentes, su planificación y resultados.

Después del análisis de las propuestas anteriores se decide que no es conveniente adoptar ninguno de estos sistemas de información para la educación con sus bases de datos y sus capas de acceso a dato, debido a que son ineficientes dadas las necesidades actuales de la Dirección de Educación de la Administración Provincial de Artemisa, pues contienen términos, clasificaciones y automatiza procesos que no son los mismos que allí se desarrollan, por lo tanto se hace necesario el desarrollo de una base de datos que permita informatizar todos los procesos que están vinculados a la gestión de la información en dicha Dirección .

1.3 Metodologías de desarrollo de *software*

Todo desarrollo de *software* es difícil de controlar, de ahí la importancia de la utilización de una metodología que te guie paso a paso con las actividades que deben desarrollarse, las personas involucradas en la realización de las mismas, así como el papel que desempeñan estas en aras de lograr un proceso de *software* exitoso. Antes de elegir la metodología que se usará para la implementación de un *software* se debe analizar cuál sería la más factible para su desarrollo.

La metodología es un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar nuevo *software*.

Las metodologías se pueden clasificar en dos grandes grupos:

- ✓ **Metodologías Pesadas:** esta metodología está orientada al control de los procesos, estableciendo rigurosamente las actividades a desarrollar, herramientas a utilizar y notaciones que se usarán.(9)

Entre las metodologías tradicionales o pesadas se pueden citar:

- ✓ RUP (*Rational Unified Procces*).
- ✓ MSF (*Microsoft Solution Framework*).

Capítulo 1: Fundamentación Teórica.

- ✓ Win-Win *Spiral Model*.
- ✓ Iconix.
- ✓ **Metodologías Ligeras/Ágiles:** esta metodología está orientada a la interacción con el cliente y el desarrollo incremental del *software*, mostrando versiones parcialmente funcionales del *software* al cliente en intervalos cortos de tiempo, para que pueda evaluar y sugerir cambios en el producto según se va desarrollando.(9)

Las metodologías Ligeras/Ágiles más destacadas hasta el momento en el desarrollo del *software* son:

- ✓ XP (*Extreme Programming*).
- ✓ SXP
- ✓ *Scrum*.
- ✓ *Crystal Clear*.
- ✓ DSDM (*Dynamic Systems Development Method*).
- ✓ FDD (*Feature Driven Development*).
- ✓ ASD (*Adaptive Software Development*).
- ✓ XBreed.
- ✓ Extreme Modeling.

A continuación se describen dos de las metodologías más utilizadas en el desarrollo de *software*:

Extreme Programming (XP)

XP se clasifica como Metodología Ágil y es una de las más exitosas en la actualidad, utilizada para proyectos de corto plazo, equipo pequeño y cuyo plazo de entrega es en tiempo récord. La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto. (10)

Capítulo 1: Fundamentación Teórica.

El ciclo de vida ideal de XP está compuesto por seis fases que permiten todo el proceso de desarrollo en un tiempo record, estas fases son las siguientes:

- ✓ Exploración.
- ✓ Planificación de la entrega.
- ✓ Iteraciones.
- ✓ Producción.
- ✓ Mantenimiento.
- ✓ Muerte del proyecto. **(11)**

Metodología SXP

SXP es una metodología de desarrollo de *software* compuesta por las metodologías SCRUM y XP que ofrece una estrategia tecnológica, a partir de la introducción de procedimientos ágiles que permitan actualizar los procesos de *software* para el mejoramiento de la actividad productiva fomentando el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo, ayudando al líder del proyecto a tener un mejor control del mismo. SCRUM es una forma de gestionar un equipo para que trabaje eficientemente y tenga siempre medidos los progresos. XP más bien es una metodología encaminada para el desarrollo; consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto.

Consta de 4 fases principales:

- ✓ Planificación-Definición donde se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto.
- ✓ Desarrollo, es donde se realiza la implementación del sistema hasta que esté listo para ser entregado.

Capítulo 1: Fundamentación Teórica.

- ✓ Entrega, puesta en marcha.
- ✓ Mantenimiento, donde se realiza el soporte para el cliente.

SXP está especialmente indicada para proyectos de pequeños equipos de trabajo, rápido cambio de requisitos o requisitos imprecisos, muy cambiantes, donde existe un alto riesgo técnico y se orienta a una entrega rápida de resultados y una alta flexibilidad. Ayuda a que trabajen todos juntos, en la misma dirección, con un objetivo claro, permitiendo además seguir de forma clara el avance de las tareas a realizar, de forma que los jefes pueden ver día a día cómo progresa el trabajo. **(11)**

Por todos los beneficios explicados anteriormente, por ser principalmente adaptable al desarrollo de esta investigación y por ser la metodología adoptada por el departamento, se utiliza SXP para la realización de este trabajo.

1.4 Introducción a las Bases de Datos

Las bases de datos se han constituido como una de las herramientas más ampliamente difundidas en la actual sociedad de la información, utilizadas como fuentes secundarias en cuanto recuperación y almacenamiento de información en todos los campos a nivel científico, social, económico, político y cultural.

Base de Datos: Una base de datos es un “almacén” que permite guardar grandes cantidades de información de forma organizada para que luego se pueda encontrar y utilizar fácilmente.

Desde el punto de vista informático, la base de datos es un sistema formado por un conjunto de datos almacenados en discos que permiten el acceso directo a ellos y un conjunto de programas que manipulen ese conjunto de datos.**(12)**

Cada base de datos se compone de una o más tablas que guarda un conjunto de datos. Cada tabla tiene una o más columnas y filas. Las columnas guardan una parte de la información sobre cada elemento que se quiera guardar en la tabla, cada fila de la tabla conforma un registro.

Características:

Capítulo 1: Fundamentación Teórica.

Entre las principales características de los sistemas de base de datos se pueden mencionar:

- Independencia lógica y física de los datos.
- Redundancia mínima.
- Acceso concurrente por parte de múltiples usuarios.
- Integridad de los datos.
- Consultas complejas optimizadas.
- Seguridad de acceso y auditoría.
- Respaldo y recuperación.
- Acceso a través de lenguajes de programación estándar.

Una base de datos cuenta con varios componentes:

- ✓ *Hardware*: Se refieren a los dispositivos de almacenamiento donde se encuentra la base de datos, así como los periféricos asociados para su utilización como son los controladores de dispositivos, unidades de discos entre otros.
- ✓ *Software*: Está constituido por un conjunto de programas que se conoce como Sistema Gestor de Base de Datos (SGBD) que es el encargado de manejar cada una de las solicitudes realizadas por los usuarios.
- ✓ *Usuario*: Existen tres grandes grupos de usuarios relacionados con las bases de datos.
- ✓ *Programador de aplicaciones*: se encarga de crear programas de aplicación que se utilizan en la base de datos.
- ✓ *Usuarios finales*: acceden a la base de datos por medio de un lenguaje de consultas o un programa de aplicaciones.

Capítulo 1: Fundamentación Teórica.

- ✓ Administrador de la base de datos: es el encargado del control general del SGBD.

Arquitectura de una base de datos

La arquitectura se divide en tres niveles generales: interno, conceptual y externo.

- Nivel Interno: es el más cercano al almacenamiento físico, es decir, el que concierne a la manera como los datos se almacenan en realidad.
- Nivel Externo: es el más cercano a los usuarios, es decir, el que atañe a la manera cómo cada usuario ve los datos.
- Nivel Conceptual: es un nivel de mediación entre los otros dos.(13)

Ver Anexo#1: Arquitectura de las Bases de Datos.

Clasificación de las bases de datos

Las bases de datos se pueden clasificar según el modo en que estas almacenan la información o sus características. Algunas de estas clasificaciones se muestran a continuación.

Según la variabilidad de los datos almacenados

Bases de datos estáticas:

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

Bases de datos dinámicas:

Éstas son bases de datos donde la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización, borrado y adición de datos, además de las operaciones fundamentales de consulta. Un ejemplo de esto puede ser la base de datos utilizada en un sistema de información de una tienda de abarrotes, una farmacia, un videoclub.

Según el contenido que almacenan

Capítulo 1: *Fundamentación Teórica.*

Bases de datos bibliográficas:

Solo contienen un subrogante (representante) de la fuente primaria, que permite localizarla. Un registro típico de una base de datos bibliográfica contiene información sobre el autor, fecha de publicación, editorial, título, edición, de una determinada publicación, entre otros. Puede contener un resumen o extracto de la publicación original, pero nunca el texto completo, porque si no, se estaría en presencia de una base de datos a texto completo (o de fuentes primarias —ver más abajo). Como su nombre lo indica, el contenido son cifras o números. Por ejemplo, una colección de resultados de análisis de laboratorio, entre otras.

Bases de datos de texto completo:

Almacenan las fuentes primarias, como por ejemplo, todo el contenido de todas las ediciones de una colección de revistas científicas.

Directorios:

Un ejemplo son las guías telefónicas en formato electrónico.

Bases de datos o "bibliotecas" de información química o biológica.

Son bases de datos que almacenan diferentes tipos de información proveniente de la química, las ciencias de la vida o médicas. Se pueden considerar en varios subtipos:

- Las que almacenan secuencias de nucleótidos o proteínas.
- Las bases de datos de rutas metabólicas.
- Bases de datos de estructura, comprende los registros de datos experimentales sobre estructuras 3D de biomoléculas.
- Bases de datos clínicas.

Fases de diseño de bases de datos

Ver Anexo#2: Fases del diseño de una Base de Datos.

1. Análisis de requerimiento:

- ◆ Captar los requisitos de información de los distintos grupos de usuarios.
- ◆ Información sobre el uso que se piensa dar a la BD.

Capítulo 1: *Fundamentación Teórica.*

◆ Captar requerimientos operativos

→ Transacciones (críticas y no críticas)

2. Diseño conceptual:

◆ Obtener una buena representación de los recursos de información de la empresa, con independencia de usuarios o aplicaciones en particular, y fuera de consideraciones sobre eficiencia del ordenador.

3. Diseño Lógico:

◆ Transformar el esquema conceptual obtenido en la etapa anterior, adaptándolo al modelo de datos en el que se apoya el SGBD que se va a utilizar.

4. Diseño Físico:

◆ Trata de conseguir una instrumentación lo más eficiente posible, del esquema lógico. **(14)**

Modelos de bases de datos

Es una colección de herramientas conceptuales para describir los datos, las relaciones que existen entre ellos, semántica asociada a los datos y restricciones de consistencia.

Los modelos de datos se dividen en tres grupos:

1. Modelos lógicos basados en registros.
2. Modelos lógicos basados en objetos.
3. Modelos físicos de datos.

Modelos lógicos basados en registros

△ Modelo de datos jerárquico:

Este modelo utiliza árboles para la representación lógica de los datos. Este árbol está compuesto de unos elementos llamados nodos. El nivel más alto del árbol se denomina raíz. Cada nodo representa un registro con sus correspondientes campos.

La representación gráfica de este modelo se realiza mediante la creación de un árbol invertido, los diferentes niveles quedan unidos mediante relaciones.

Capítulo 1: Fundamentación Teórica.

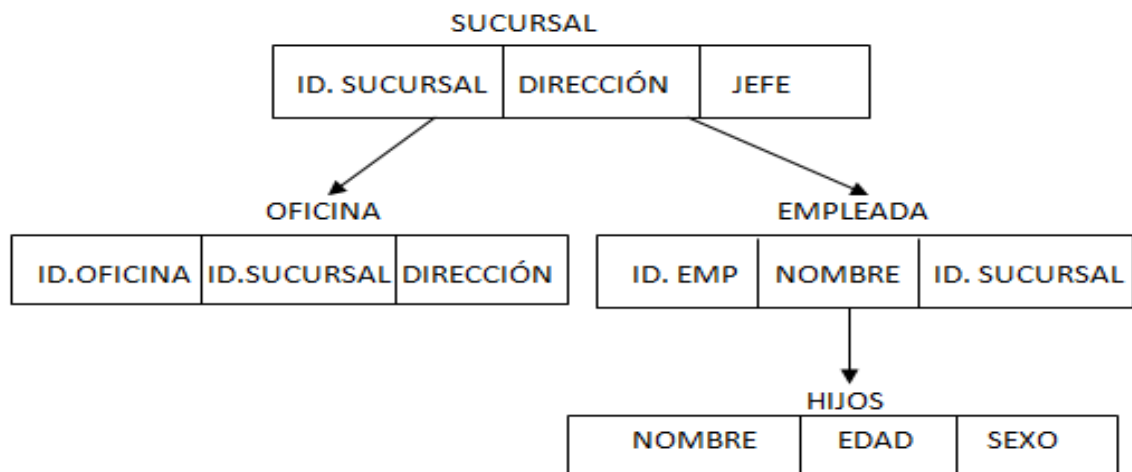


Figura 1 Árbol invertido.

En este modelo solo se pueden representar relaciones 1: M, por lo que presenta varios inconvenientes:

1. No se admiten relaciones N: M
2. Un segmento hijo no puede tener más de un padre.
3. No se permiten más de una relación entre dos segmentos.
4. Para acceder a cualquier segmento es necesario comenzar por el segmento raíz
5. El árbol se debe de recorrer en el orden designado.

Modelo de datos en red:

En este modelo las entidades se representan como nodos y sus relaciones son las líneas que los unen. En esta estructura cualquier componente puede relacionarse con cualquier otro.

A diferencia del modelo jerárquico, en este modelo, un hijo puede tener varios padres.

Los conceptos básicos en el modelo en red son:

- El tipo de registro, que representa un nodo.

Capítulo 1: Fundamentación Teórica.

- Elemento, que es un campo de datos.
- Agregado de datos, que define un conjunto de datos con nombre.

Este modelo de datos permite representar relaciones N: M

Modelo de datos relacional:

Este modelo es el más utilizado actualmente ya que utiliza tablas bidimensionales para la representación lógica de los datos y sus relaciones.

Algunas de sus principales características son:

- Puede ser entendido y usado por cualquier usuario.
- Permite ampliar el esquema conceptual sin modificar las aplicaciones de gestión.
- Los usuarios no necesitan saber donde se encuentran los datos físicamente.

El elemento principal de este modelo es la relación que se representa mediante una tabla.

Modelos lógicos basados en registros.

Se utilizan para describir los datos en los niveles conceptual y físico. Además, para especificar la estructura lógica global de la base de datos y para proporcionar una descripción al nivel más alto de la implementación.

Modelos lógicos basados en objetos.

Se usan para describir datos en los niveles conceptuales y de visión, es decir, con este modelo se representan los datos de tal forma que se puedan captar en el mundo real, tienen una capacidad de estructuración bastante flexible y permiten especificar restricciones de datos explícitamente. Existen diferentes modelos de este tipo, pero el más utilizado por su sencillez y eficiencia es el modelo Entidad-Relación.

Modelo Entidad-Relación.

Capítulo 1: Fundamentación Teórica.

Denominado por sus siglas como: E-R; Este modelo representa a la realidad a través de *entidades*, que son objetos que existen y que se distinguen de otros por sus características, por ejemplo: un alumno se distingue de otro por sus características particulares como lo es el nombre, o el número de control asignado al entrar a una institución educativa, así mismo, un empleado, una materia, entre otros. Las entidades pueden ser de dos tipos:

Tangibles:

Son todos aquellos objetos físicos que se pueden ver, tocar o sentir.

Intangibles:

Todos aquellos eventos u objetos conceptuales que no se pueden ver, aun sabiendo que existen, por ejemplo: la entidad materia, se sabe que existe, sin embargo, no se puede visualizar o tocar.

Las características de las entidades en base de datos se llaman atributos, por ejemplo el nombre, dirección teléfono, grado y grupo, son atributos de la entidad alumno; Clave, número de seguro social y departamento, son atributos de la entidad empleado. A su vez una entidad se puede asociar o relacionar con más entidades a través de relaciones.

Modelos físicos de datos

Se usan para describir a los datos en el nivel más bajo, aunque existen muy pocos modelos de este tipo, básicamente capturan aspectos de la implementación de los sistemas de base de datos. Existen dos clasificaciones de este tipo que son:

- Modelo unificador
- Memoria de elementos. **(15)**

1.5 Herramientas y Tecnologías utilizadas

1.5.1 Sistema Gestor de Bases de Datos (SGBD)

Existen varias definiciones sobre los SGBD, algunas se muestran a continuación:

Capítulo 1: Fundamentación Teórica.

- Un Sistema de Gestión de Bases de Datos consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a esos datos. El Objetivo primordial de un SGBD es proporcionar un entorno que sea a la vez conveniente y eficiente para ser utilizado al extraer y almacenar información de la base de datos **(16)**
- “El sistema de gestión de la base de datos es una aplicación que permite a los usuarios definir, crear y mantener la base de datos, y proporciona acceso controlado a la misma.” **(17)**

Un SGBD no es más que un tipo de aplicación informática o *software* que actúa como interfaz entre el usuario y la base de datos. Funciona como un intermediario para que el usuario pueda comunicarse con la base de datos en términos abstractos y de una forma práctica y eficiente, abstrayéndolo del modo físico de almacenamiento de la información en la computadora, y del método de acceso empleado, permitiéndole definir, crear y mantener la base de datos, y proporcionándole el acceso controlado a la misma.

Sistema Gestor de Bases de Datos PostgreSQL

Es un sistema de bases de datos relacional de *software* libre que está catalogado como la mejor y más avanzada base de datos Open Source del mundo. Iniciado como un proyecto universitario en la década de 1980, ha llevado varios períodos de desarrollo y cuenta con una arquitectura confiable, estabilidad de procesamiento e integridad en el manejo de datos. Es un verdadero proyecto de *software* abierto, ya que su desarrollo no es dirigido por una empresa sino que es administrado por una comunidad. PostgreSQL es un sistema relacional orientado a objetos, ya que incluye características como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Es liberado bajo la licencia BSD, la cual permite generar versiones comerciales del producto.

Capítulo 1: Fundamentación Teórica.

Ver Anexo#3. Arquitectura de PostgreSQL.

Características:

- MVCC: Control de Concurrencia Multi-Versión (*Multi-Version Concurrency Control*), es la tecnología que usa para evitar bloqueos innecesarios.
- Multiplataforma.
- Soporta claves foráneas.
- Posee buenas interfaces de instalación y administración.
- *Write Ahead Logging* (Escritura anticipada del registro): incrementa la dependencia de la base de datos al registro de cambios antes de que estos sean escritos en la base de datos. Esto garantiza que en el hipotético caso de que la base de datos se desconecte, existirá un registro de las transacciones a partir del cual se pueda restaurar la base de datos.
- Cumple con las propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad).
- Ha sido diseñado para mantenerse estable con grandes volúmenes de datos y transacciones.
- Es altamente configurable, con lo cual puede personalizarse de acuerdo al escenario donde será utilizado. **(18)**

Herramienta de administración y desarrollo para PostgreSQL

Para la administración y desarrollo con PostgreSQL se utilizará la herramienta pgAdmin III, esta es una aplicación gráfica para gestionar PostgreSQL, siendo la más completa y popular con licencia Open Source. Está escrita en C++ usando la librería gráfica multiplataforma *wxWidgets*, lo que permite que se pueda usar en Linux, FreeBSD, Mac OS X y Windows. Es capaz de gestionar versiones a partir de la de PostgreSQL 7.3 ejecutándose en cualquier plataforma. Está diseñada para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y facilita enormemente la administración. La conexión al servidor puede hacerse mediante el protocolo TCP/IP.

Capítulo 1: Fundamentación Teórica.

1.5.2 Frameworks

En el desarrollo de *software*, Marco de trabajo o Framework en su término en inglés, es una estructura de soporte definida en la cual otro proyecto de *software* puede ser organizado y desarrollado. Típicamente, un *framework* puede incluir soporte de programas, bibliotecas y un lenguaje interpretado, entre otros *software* para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Define una arquitectura adaptada a las particularidades de un determinado dominio de aplicación, definiendo de forma abstracta una serie de componentes y sus interfaces, y estableciendo las reglas y mecanismos de interacción entre ellos.

El interés en reutilizar *software* ha sido cambiado de la reutilización de componentes simples a diseño de sistemas enteros o estructuras de aplicaciones. Un sistema *software* que pudiera ser reutilizado en este nivel para la creación de aplicaciones completas es llamado *framework*. Los *frameworks* están basados en la idea que deberían permitir la producción fácil de un conjunto de sistemas específicos pero similares, dentro de un cierto dominio comenzando desde una estructura genérica. Brevemente, los *frameworks* son arquitecturas genéricas integradas por un extensible conjunto de componentes. Además, los *frameworks* pueden contener *subframeworks* los cuales representen subconjuntos de componentes de un sistema más grande.

Framework Hibernate

Hibernate es una herramienta de Mapeo objeto-relacional (ORM) para la plataforma Java (y disponible también para .Net con el nombre de NHibernate) que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) o anotaciones en los *beans* de las entidades que permiten establecer estas relaciones.

Como todas las herramientas de su tipo, *Hibernate* busca solucionar el problema de la diferencia entre los dos modelos de datos coexistentes en una aplicación: el usado en la memoria de la computadora (orientación a objetos) y el usado en las

Capítulo 1: Fundamentación Teórica.

bases de datos (modelo relacional). Para lograr esto permite al desarrollador detallar cómo es su modelo de datos, qué relaciones existen y qué forma tienen. Con esta información Hibernate le permite a la aplicación manipular los datos de la base operando sobre objetos, con todas las características de la Programación Orientado a Objetos (POO). Hibernate convertirá los datos entre los tipos utilizados por Java y los definidos por SQL. Hibernate genera las sentencias SQL y libera al desarrollador del manejo manual de los datos que resultan de la ejecución de dichas sentencias, manteniendo la portabilidad entre todos los motores de bases de datos con un ligero incremento en el tiempo de ejecución.

Hibernate es *software* libre, distribuido bajo los términos de la licencia GNU LGPL. (19)

Spring 3.0

Spring es un framework de aplicación desarrollado por la compañía Interface, para aplicaciones escritas en el lenguaje de programación Java, es ligero ya que no es una aplicación que requiere mucho para su ejecución.

Spring es un framework de código abierto, por su diseño ofrece mucha libertad a los desarrolladores en Java y soluciones muy bien documentadas y fáciles de usar. Está diseñado como una serie de módulos que pueden trabajar independientemente uno de otro, intenta mantener un mínimo acoplamiento entre la aplicación y el propio framework de forma que podría ser desvinculada de él sin demasiada dificultad. Es potente en cuanto a la gestión del ciclo de vida de los componentes y fácilmente ampliable.

1.5.3 Herramientas CASE (*Computer Aided Software Engineering*)

Estas herramientas permiten el diseño de un *software* antes de su codificación, esto se realiza mediante diagramas que permiten que todo el equipo de desarrollo comprenda lo que se quiere hacer realmente y que el trabajo que se realice sea de forma homogénea. Algunas de estas herramientas tienen un valor económico muy alto y requieren altos costos de entrenamiento para el personal. Por otra parte,

Capítulo 1: Fundamentación Teórica.

algunas herramientas CASE no ofrecen o evalúan soluciones potenciales para los problemas relacionados con sistemas, o simplemente no llevan a cabo ningún análisis de los requerimientos de la aplicación. Entre las principales herramientas se encuentran:

Embarcadero ER / Studio

Es una herramienta de modelado de datos fácil de usar para el diseño y construcción de bases de datos a nivel físico y lógico. Direcciona las necesidades diarias de los administradores, desarrolladores y arquitectos que construyen y mantienen aplicaciones de bases de datos grandes y complejas.

Principales funcionalidades:

- Capacidad fuerte en el diseño lógico.
- Construcción automática de base de datos.
- Ingeniería inversa de base de datos.
- Un Repositorio para el modelado.

Documentación basada en HTML.

Soporta el muy popular SQL y base de datos de escritorio, incluyendo: Oracle, Microsoft Access, Microsoft SQL Server entre otras.

Visual Paradigm

Es una herramienta CASE que utiliza UML como lenguaje de modelado, soporta el diagrama entidad-relación. Está disponible en varias ediciones: Enterprise, Professional, Community, Standard, Modeler y Personal, que permite escoger la más idónea para el usuario según su necesidad. Esta herramienta se distribuye bajo licencia gratuita y comercial, entre las principales características que presenta Visual Paradigm están: producto de calidad, multiplataforma, soporta aplicaciones web, varios idiomas, fácil de instalar y actualizar, compatibilidad entre ediciones, soporta a miles de usuarios trabajando sobre el mismo proyecto y permite que cada uno vea los cambios realizados en tiempo real. Esta herramienta tiene habilitado UML 2.1, un estándar ampliamente utilizado actualmente en las empresas para el

Capítulo 1: Fundamentación Teórica.

modelado de *software*. Permite obtener el diagrama Entidad-Relación a partir del diagrama de clases persistentes o viceversa. Tiene la capacidad de crear el esquema de clases a partir de una base de datos y crear la base de datos a partir del esquema de clases. A partir de un modelo relacional en SQL Server, MySQL u otro, es capaz de desplegar todas las clases asociadas a las tablas. Posibilita la generación de bases de datos y la transformación de diagramas de Entidad-Relación en tablas de base de datos. Además, la documentación es generada en varios formatos como JPG, XML, Excel y en el caso de la base de datos genera un script con las tablas.

◆ **Herramienta CASE seleccionada**

A partir de un análisis realizado a las principales herramientas CASE anteriormente expuestas se concluyó que para el Modelado de la Base de Datos para el grupo de la Dirección de Educación se utilizará el Visual Paradigm UML en su versión 6.4. Este *software* brinda múltiples ventajas al usuario como son: es fácil de instalar y actualizar, es un *software* multiplataforma. Visual Paradigm permite realizar todo tipo de diagramas de clases, generar código desde diagramas y generar documentación. Además, facilita la generación de bases de datos, transformación de diagramas de Entidad-Relación en tablas de base de datos e ingeniería inversa.

(20)

1.5.4 Netbeans 7.01

Es un entorno de desarrollo integrado (IDE), hecho principalmente para el lenguaje de programación Java. NetBeans IDE es un producto libre y gratuito sin restricciones de uso. NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo. Sun Microsystems fundó el proyecto de código abierto en junio de 2000 y continúa siendo el patrocinador principal de los proyectos. La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de *software* llamados

Capítulo 1: Fundamentación Teórica.

módulos. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las APIs de NetBeans y un archivo especial (*manifest file*) que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de *software*.

1.5.5 Herramientas de control de versiones

Las herramientas de control de versiones, facilitan el control, revisión o edición de las distintas versiones que pueda tener un producto en desarrollo o culminado.

Subversion (SVN)

El sistema de control de versiones es un software libre que administra el acceso a un conjunto de ficheros y mantiene una historia de cambios realizados. El control de versiones es útil para guardar cualquier documento que se cambie con frecuencia. Consiste en una copia muestra en un repositorio central y un programa cliente con el que cada usuario sincroniza su copia local, lo que permite compartir los cambios sobre un mismo conjunto de ficheros. El repositorio guarda registro de los cambios realizados por cada usuario, y permite volver a un estado anterior en caso de necesidad.

Ventajas

- ✓ Actualización de ficheros modificados.
- ✓ Copias de seguridad centralizadas.
- ✓ Historial de Cambios.
- ✓ Brinda acceso remoto.
- ✓ Provee seguridad al sistema.

RapidSVN

Se utiliza para gestionar los datos del repositorio SVN. Es una herramienta libre que permite darle a cada integrante del proyecto diferentes permisos según los documentos y la información que sea necesaria para el rol que desempeñe.

Capítulo 1: Fundamentación Teórica.

Entre las principales características se destacan:

Simple - proporciona una interfaz fácil de usar para las características de Subversion.

Eficiente - simple para los principiantes pero lo suficientemente flexible como para aumentar la productividad para los usuarios de Subversion con experiencia.

Portable - se ejecuta en cualquier plataforma en la que Subversion y wxWidgets puede ejecutar: Linux, Windows, Mac OS / X, Solaris, entre otros.

Rápido - completamente escrito en C ++.

Multilingüe - que ha sido traducido a muchos idiomas ya: alemán, francés, italiano, portugués, ruso, ucraniano, chino simplificado, japonés.

Conclusiones

Los SGBD han evolucionado considerablemente desde sus inicios, mejorando cada vez más los servicios que brindan, lográndose con ellos que la información sea almacenada de forma estructurada y no duplicada. Además, que garantizan el acceso controlado a la información.

En el mundo existen diversas herramientas para el diseño y procesamiento de las bases de datos, las que se van ampliando y perfeccionando con el tiempo. Definir cuál de ellas utilizar es cada vez más difícil. Por tal motivo en este capítulo se analizaron los principales modelos de bases de datos que ayuden al diseño eficiente de la misma, los gestores más utilizados en el mundo y en la UCI, y las herramientas de modelado que ayuden en este trabajo.

Capítulo 2: Diseño y Arquitectura de la base de datos.

Capítulo 2: Diseño y Arquitectura de la base de datos

Introducción

En este capítulo se brinda información acerca de las técnicas de diseño de las Bases de Datos. Se exponen aspectos relativos acerca de la arquitectura de la capa de acceso a dato, lo que da lugar a los requisitos funcionales de la misma. Se describen los patrones de diseño a utilizar y se definen los artefactos correspondientes al rol diseñador de BD, dígame artefactos, Modelo Entidad Relación (MER), Modelo Relacional (MR). Se muestra el modelo físico que componen la BD, el MER de la Dirección de Educación, y la descripción de alguna de sus tablas y la transformación de este al MR.

2.1 Metodología para el diseño de bases de datos.

La metodología para el diseño de la BD, consta de los siguientes pasos:

- ✓ Determinación de entidades y atributos.
- ✓ Normalización de entidades.
- ✓ Determinación de relaciones.
- ✓ Obtención del modelo lógico global de los datos.
- ✓ Diseño físico de la BD.

Cuando se realiza el diseño de la BD para un sistema determinado, es necesario establecer los datos a tomar en cuenta y las dependencias funcionales existentes entre ellos.

2.2 Técnicas para diseñar bases de datos relacionales

Existen dos escuelas que implican dos grupos de algoritmos a la hora de aplicar la Teoría de Normalización: los métodos de Análisis o descomposición y los Procedimientos de síntesis, que a continuación se detallan.

Capítulo 2: Diseño y Arquitectura de la base de datos.

Diseño descendente, Análisis o descomposición

Este método es el que se emplea mediante un proceso denominado “normalización”. El mismo parte de una relación llamada Universal que contiene todos los atributos del universo del discurso y las dependencias funcionales existentes entre ellas. Por medio de descomposiciones sucesivas y cumpliendo determinados principios de conservación de la información y de las dependencias, resultan esquemas de menor grado en formas normales cada vez más avanzadas, por lo que se puede garantizar que se reducen las anomalías. Se pueden aplicar estos métodos también a cada una de las relaciones obtenidas a partir de un esquema conceptual en un modelo de datos de alto nivel (Modelo Entidad-Relación) y luego transformar el esquema conceptual a un conjunto de relaciones, empleando un procedimiento de transformación. También se puede aplicar el método a cada una de las relaciones que se obtienen de la transformación del Diagrama Entidad-Relación al modelo relacional. El objetivo inicial que se pretende con estos métodos es separar la información referente a un objeto o entidad diferente.

Síntesis relacional o Procedimientos de síntesis

Este es un método de diseño alternativo a la descomposición, resulta ser un enfoque más purista. Implica contemplar el diseño de esquemas de BD relacionales estrictamente en términos de dependencias funcionales, y otros tipos especificados para los atributos de la BD. Aquí los esquemas de relación en tercera forma normal o forma normal de *Boyce-Codd* se sintetizan gradualmente agrupando los atributos apropiados. A partir de conjuntos de atributos y dependencias funcionales se obtienen relaciones. Recorre un camino inverso a la descomposición, es decir, busca agrupar atributos, a fin de tener en una relación toda la información correspondiente a un objeto o entidad **(21)**.

2.2.1 Normalización

El concepto de normalización fue introducido por E.D. Codd y fue concebido para aplicarse a sistemas relacionales. Sin embargo, tiene aplicaciones más amplias. La

Capítulo 2: Diseño y Arquitectura de la base de datos.

normalización es el proceso mediante el cual se transforman datos complejos a un conjunto de estructuras de datos más pequeñas, que además de ser más simples y más estables, son más fáciles de mantener. También se puede entender la normalización como una serie de reglas que sirven para ayudar a los diseñadores de bases de datos a desarrollar un esquema que minimice los problemas de lógica. Cada regla está basada en la que le antecede. La normalización se adoptó porque el viejo estilo de poner todos los datos en un solo lugar, como un archivo o una tabla de la base de datos, era ineficiente y conducía a errores de lógica cuando se trataban de manipular los datos.(22)

Ver Anexo#4: Diagrama de inclusión de todas las formas normales.

La normalización involucra varias fases que se realizan en orden. La realización de la segunda fase supone que se ha concluido la primera y así sucesivamente. Tras completar cada fase se dice que la relación está en:

- Primera Forma Normal (1FN) una relación está en dicha forma normal si los valores de sus atributos son atómicos, los dominios no tienen elementos que a su vez sean conjuntos y la relación, no incluye ningún grupo repetitivo.
- Segunda Forma Normal (2FN) debe estar en 1FN y además define que los atributos no llaves son funcionales y completamente dependientes de la llave primaria, por lo general compuesta. Asegura que todas las columnas que no son llave sean completamente dependientes de la llave primaria (PK).
- Tercera Forma Normal (3FN) debe de estar en 2FN y señala que no puede existir cualquier dependencia transitiva. Una dependencia transitiva es aquella en la cual las columnas que no son llave son dependientes de otras columnas que tampoco son llave.
- Forma Normal de Boyce-Codd (FNBC) debe cumplir con las 3FN anteriores, además satisface que cada determinante es una llave (candidata o primaria).
- Cuarta Forma Normal (4FN): debe estar en FNBC y que además existan tres o más atributos formando parte de la llave primaria. Debe cumplir que no existan

Capítulo 2: Diseño y Arquitectura de la base de datos.

dos o más atributos independientes con dependencia respecto a un conjunto de atributos, formando parte de la llave junto a dicho conjunto, es decir, no dependencia entre atributos llaves.

- Quinta Forma Normal (5FN): una tabla se encuentra en 5FN si la tabla está en 4FN y no existen relaciones de dependencias no triviales que no siguen los criterios de las claves. Una tabla que se encuentra en la 4FN se dice que está en la 5FN si, cada relación de dependencia se encuentra definida por las claves candidatas. **(22)**

2.3 Requisitos Funcionales y no Funcionales del Módulo de la Dirección de Educación.

Requisitos Funcionales: Son las capacidades o condiciones que el sistema debe cumplir. Los requerimientos funcionales se mantienen invariables sin importar con qué propiedades o cualidades se relacionen.

A continuación se muestran algunos requisitos funcionales del sistema, para más información dirigirse a la Plantilla lista de reserva del producto (LRP):

Tabla 1 Requisitos Funcionales.

RF1	Insertar información sobre la situación estadística de la enseñanza especial.
RF2	Modificar información sobre la situación estadística de la enseñanza especial.
RF3	Eliminar información sobre la situación estadística de la enseñanza especial.
RF4	Buscar información sobre la situación estadística de la enseñanza especial.
RF5	Insertar información sobre las modalidades de la enseñanza especial.
RF6	Modificar información sobre las modalidades de la enseñanza especial.

Capítulo 2: Diseño y Arquitectura de la base de datos.

- RF7 Eliminar información sobre las modalidades de la enseñanza especial.
- RF8 Buscar información sobre las modalidades de la enseñanza especial.
- RF9 Insertar información sobre la proyección del egreso en la enseñanza especial.
- RF10 Modificar información sobre la proyección del egreso en la enseñanza especial.
- RF11 Eliminar información sobre la proyección del egreso en la enseñanza especial.
- RF12 Buscar información sobre la proyección del egreso en la enseñanza especial.

Requisitos no Funcionales: Son las propiedades o cualidades que el producto debe tener, debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable.

RNF 1 Restricciones de Diseño

- 1.1 Como Sistema Gestor de Base de Datos se utilizará PostgreSQL en su versión 9.0 o superior.
- 1.2 Las herramientas CASE que se utilizaron el Visual Paradigm para UML en su versión 6.4 y Power Architect en su versión 0.9.13 para el diseño del modelo físico de los datos y la generación de otros artefactos.
- 1.3 Si se intenta acceder al sistema desde una estación no autorizada, el acceso será denegado.

RNF 2 Soporte

- 2.1 Realizar pruebas, mantenimiento e instalaciones necesarias para lograr el mejoramiento y evolución en el tiempo.

RNF 3 Usabilidad

Capítulo 2: Diseño y Arquitectura de la base de datos.

3.1 La Base de Datos con su capa de acceso brindará gran confiabilidad y centralidad en cuanto a la información que se desee administrar en la base de datos.

RNF 4 Seguridad

4.1 La información que se maneje en el sistema estará protegida de acceso no autorizado y divulgación, a partir de los diferentes roles de los usuarios que empleen en la BD.

4.2 El Sistema Gestor de Base de Datos escogido debe presentar facilidades de administración de roles y usuarios restringiendo el acceso a los datos.

RNF 5 Confiabilidad

5.1 Deben establecerse los mecanismos necesarios para el restablecimiento de la base de datos ante fallos de comunicación u otros.

5.2 Deben garantizarse la realización de copias de respaldo de todos los datos periódicamente.

2.4 Diseño e implementación de la Capa de Acceso a Datos.

2.4.1 Patrones de diseño utilizados.

Un patrón describe un problema que ocurre varias veces, además de plantear la solución a este problema pudiendo utilizarse tantas veces sea necesario.

Los patrones de diseño son principios generales de soluciones que aplican ciertos estilos que ayudan a la creación de *software*. Es una descripción de un problema y la solución a la que le da el nombre y que se puede aplicar en nuevos contextos. Muchos patrones ayudan a asignar responsabilidades a los objetos. **(23)**

Los patrones de diseño son soluciones simples a problemas específicos y comunes del diseño orientado a objetos, su principal objetivo es agrupar una colección de soluciones de diseño que sean válidas en distintos contextos. Es una solución a un problema de diseño no trivial que es efectiva, además facilitan el aprendizaje al programador inexperto, pudiendo establecer parejas problema-solución. **(24)**

Capítulo 2: Diseño y Arquitectura de la base de datos.

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de *software*. Brindan una solución ya probada y documentada a problemas de desarrollo de *software* que están sujetos a contextos similares.

Se debe tener presente los siguientes elementos de un patrón: su nombre, el problema (cuando aplicar un patrón), la solución (descripción abstracta del problema) y las consecuencias (costos y beneficios). **(25)**

Se utilizaron varios patrones de diseño dentro de los que se encuentran los patrones descritos por Kyle Brown y Bruce G. Whitenack, a continuación se describe la utilización de los mismos.

Representación de objetos como tablas

Problema: ¿Cómo representar un objeto en un esquema de base de datos relacionales?

Solución: Definir una tabla para cada clase de objetos persistentes. Los atributos de la clase que son tipos primitivos serán las columnas de las tablas.

Representación de relaciones como tablas

Problema: ¿Cómo representar una relación en un esquema de base de datos relacionales?

Solución:

1. Para las relaciones de uno a uno ó uno a muchos:

Colocar una clave ajena en la tabla de cardinalidad uno, para representar la relación de los objetos.

O crear una tabla asociativa para registrar los identificadores de cada uno de los objetos de la relación.

Para las relaciones muchos a muchos:

Crear una tabla asociativa para registrar los identificadores de cada uno de los objetos de la relación.

Identificador de objetos

Problema: ¿Cómo mantener la identidad de un objeto en una base de datos relacional? Cada identidad

Capítulo 2: Diseño y Arquitectura de la base de datos.

de objetos individuales debe ser presentada en la base de datos.

Solución: Asignar un identificador independiente (OID) a cada objeto persistente. Se recomienda el uso de un generador de secuencias si hay alguno disponible en la base de datos ya que los identificadores son generalmente enteros largos que garantizan que sean únicos para una clase de objetos en particular.

Referencia de llaves foráneas

Problema: ¿Cómo representar objetos que referencian otros objetos que no son de tipos de datos base?

Solución: Asignar a cada objetos un identificador único. Luego añadir una columna por cada variable de instancia que no tenga un tipo de dato base o sea una colección. En esa columna almacenar el identificador del objeto referenciado y declarar la columna como llave foránea.

Representar una herencia en una base de datos relacional

Problema: ¿Cómo representar una jerarquía de herencia en una base de datos relacional?

Solución: Crear una tabla para cada clase en la herencia que tenga atributos. Cada atributo de la clase será una columna de la tabla, añadir además una columna adicional que represente la llave común entre todas las clases de la herencia.

Otro patrón utilizado es el de claves subrogadas. Estas claves no son más que un identificador único que se asigna a cada registro de una tabla. Son de tipo numérico secuencial sin significado especial y debe ser el único campo que sea clave principal de cada tabla.

2.4.2 Situación actual de la Dirección de Educación

En la Dirección de Educación existen problemas en cuanto a la integridad, confiabilidad y centralidad de la información, es por eso que se hace necesaria la implantación de un sistema de gestión de información con su respectiva capa de acceso a dato que automatice los procesos llevados a cabo en la Dirección.

Capítulo 2: Diseño y Arquitectura de la base de datos.

El sistema de gestión de información facilitaría en gran medida el trabajo a realizar, pues este quedaría más organizado y se optimizarían los tiempos de búsqueda, así como la información a tratar puede llegar a resultar mucho más veraz al incluirse procedimientos de integridad de la información.

2.4.3 Propuesta de Solución

Partiendo de esta necesidad, se propuso por parte de la Facultad Regional “Mártires de Artemisa” desarrollar una base de datos para la Dirección de Educación con su respectiva capa de acceso a datos capaz de suplir estas dificultades.

La base de dato que se desea desarrollar para la Dirección de Educación de la Administración Provincial de Artemisa tiene como objetivo principal garantizar un correcto almacenamiento de la información, además de contribuir a la integridad, disponibilidad y centralidad de la misma.

El patrón DAO (Data Access Object) viene a resolver el problema usando un Objeto de Acceso a Datos para abstraer y encapsular el acceso a los datos. DAO es un método muy simple de mapear objetos a bases de datos. Un DAO maneja la conexión con la fuente de datos para obtener y guardar los datos, siempre realiza operaciones atómicas contra la base de datos y nunca son necesarias las transacciones. Algunos ejemplos de operaciones serían: búsquedas por una clave, creación, actualización y borrado de un registro, obtener todos los registros y cualquier otra operación que se vaya a realizar a menudo.

Normalmente se crea un DAO por cada Objeto que se use en la aplicación. Un objeto puede ser muchas cosas, lo mismo una tabla o un *Entity* de *Hibernate*. Para generar un DAO el desarrollador podría escribir una clase que contiene un atributo para cada campo de una tabla y una clase *Dao* para esa tabla que contiene los métodos para la inserción, actualización, selección y eliminación de filas.

2.4.4 Técnicas para crear Capas de Acceso a Datos.

Existen distintas formas de encarar la creación de una capa de acceso a datos:

Capítulo 2: Diseño y Arquitectura de la base de datos.

1. Una forma es crear una clase por cada tabla en la base de datos y mapearlas una a una. Esta es la forma más sencilla y es muy válida cuando se tiene una estructura de base de datos que sea muy parecida al modelo de dominio (que no se haya optimizado con particiones horizontales o uniones).

Otra forma es respetar un modelo orientado a objetos que represente los objetos del dominio del negocio. Esta es una forma más correcta de trabajar pero se hacen más complejo mapear esas clases a sus tablas correspondientes.

Para realizar esta tarea existen herramientas que se llaman ORM (*Object to Relational Mapper*).

2.5 Modelos empleados para la elaboración de la BD de la Dirección de Educación.

- **Modelo Entidad Relación.**

Un diagrama o modelo entidad-relación (a veces denominado por sus siglas, *E-R* "*Entity relation ship*", o, "DER" Diagrama de Entidad Relación) es una herramienta para el modelado de datos de un sistema de información. Estos modelos expresan entidades relevantes para un sistema de información así como sus interrelaciones y propiedades.

Estos modelos expresan entidades relevantes, así como sus interrelaciones y propiedades.

Propuesto por Peter Chen en 1976, mediante el mismo se pretenden visualizar los elementos que pertenecen a una BD, que reciben el nombre de entidades, las cuales se corresponden con el concepto de clase de la Programación Orientada a Objeto. Los cuatro elementos fundamentales de un MER son: las entidades, los atributos, las interrelaciones y el dominio.

- **Entidad:** es cualquier objeto, real o abstracto, que existe en un contexto determinado o puede llegar a existir y del cual se puede guardar información.
- **Atributos:** son características o propiedades asociadas a la entidad que toman valor en una instancia particular. Ejemplo: nombre, cédula, teléfono.

Capítulo 2: Diseño y Arquitectura de la base de datos.

- **Clave principal o primaria:** se le llama al atributo o conjunto mínimo de atributos (uno o más campos) que permiten identificar en forma única cada instancia de la entidad, es decir, a cada registro de la tabla.
- **Clave foránea:** (también llamada externa o secundaria) es un atributo que es clave primaria en otra entidad con la cual se relaciona.(26)

A continuación se mostrará un pequeño fragmento del DER de la Dirección de Educación, los demás se encuentran en el Modelo Canónico de datos de dicha dirección.

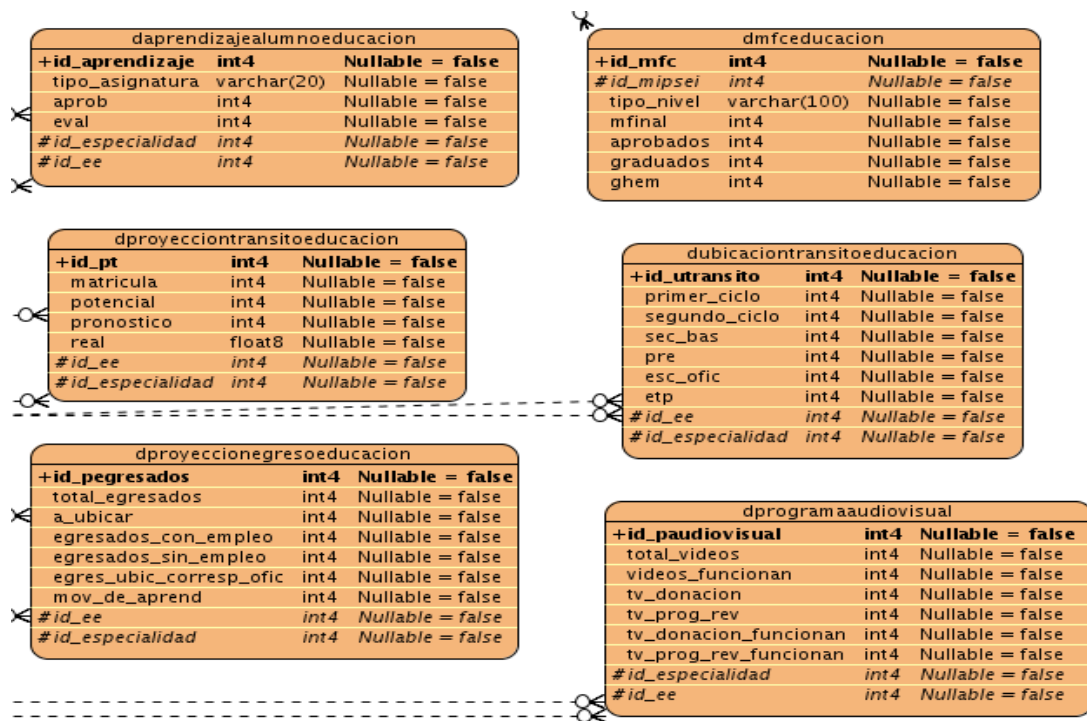


Figura 2 Diagrama Entidad-Relación.

• Descripción de las entidades

Una entidad no es más que un objeto al cual se pueden asociar atributos, es una forma física de visualizar la información. A continuación se muestra la descripción de algunas de las entidades del diseño físico, la descripción del resto de las tablas se encuentran en el Modelo Canónico de datos de dicha Dirección.

Capítulo 2: Diseño y Arquitectura de la base de datos.

Tabla 2 Descripción de la entidad del modelo.

Entidad:	dmodeloeducación		
Descripción:	Entidad que recoge por cada curso escolar Modelos de Enseñanza en la Educación Especial y Primaria.		
Relaciones:			
Campos	Tipo de Dato	Tamaño	Descripción
curso_escolar	varchar	15	Curso escolar
Provincia	varchar	10	Nombre de la provincia
Nombre	varchar	100	Nombre del modelo

Tabla 3 Descripción de la entidad Aprendizaje del alumno.

Entidad:	daprendizajealumnoeducación		
Descripción:	Entidad que guarda los datos del aprendizaje del alumno correspondiente al modelo de Educación Especial.		
Relaciones:	dmodeloeducación, nespecialidad1educacion		
Campos	Tipo de Dato	Tamaño	Descripción
id_modelo	integer	10	Identificador del modelo
id_especialidad	varchar	10	Nombre de la especialidad
tipo_asignatura	varchar	20	Nombre de la asignatura
Aprobados	integer	10	Cant de estudiantes aprobados en la asignatura
Evaluados	integer	10	Cant de estudiantes evaluados por asignatura
Fecha	date	10	Fecha actual

Capítulo 2: Diseño y Arquitectura de la base de datos.

Tabla 4 Descripción de la entidad Matricula EIDE y EVA.

Entidad:	dmatriculaEIDEyEVAeducacion		
Descripción:	Entidad que guarda una serie de campos relacionados con la Educación Primaria.		
Relaciones:	dmodeloeducación, nmunicipioeducación		
Campos	Tipo de Dato	Tamaño	Descripción
id_modelo	integer	10	Identificador del modelo
id_municipio	integer	10	Identificador del municipio
tipo_escuela	varchar	10	El nombre de la escuela ya sea EIDE o EVA
cant	integer	10	Cant pormunicipio
primero	integer	10	Cant de primero
segundo	integer	10	Cant de segundo
Tercero	integer	10	Cant de tercero
cuarto	integer	10	Cant de cuarto
quinto	integer	10	Cant de quinto
sexto	integer	10	Cant de sexto
fecha	date	10	Fecha actual
fecha_actual	timestamp	10	Fecha actual de la pc

Tabla 5 Descripción de la entidad modalidades.

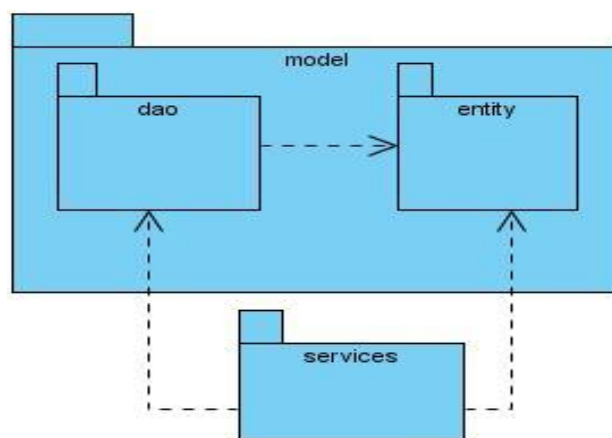
Entidad:	nmodalidadesalumnoseducacion		
Descripción:	Entidad que guarda las modalidades de la Educación Especial.		
Relaciones:			
Campos	Tipo de Dato	Tamaño	Descripción
id_modalidades	integer	10	Identificador de la modalidad
modalidades	varchar	100	Nombre de la modalidad

Capítulo 2: Diseño y Arquitectura de la base de datos.

2.6 Diagrama de Paquetes

El Diseño con metáforas es sencillamente el diseño de la solución más simple que pueda funcionar y ser implementado en un momento dado del proyecto; lo cual genera el artefacto conocido como Modelo de Diseño, que a su vez está compuesto por un diagrama de paquetes, el cual expone dicho diseño.(27)

A continuación se representa el diagrama de paquetes para el sistema que se propone.



Fig#3:

Figura 3 Diagrama de paquetes para la Dirección de Educación.

Descripción del Diagrama De Paquetes

- En el paquete **Service** es donde se encuentran los eventos correspondientes para cada componente.
- En el paquete **Model** se encuentran el paquete **Entity**, que es donde se almacenan las entidades de la base de datos y el paquete DAO, donde se ubica el DaoGenérico de la misma.

2.7 Diagrama de Componente

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Muestran las opciones de realización incluyendo código fuente, binario y

Capítulo 2: Diseño y Arquitectura de la base de datos.

ejecutable. Los componentes representan todos los tipos de elementos de *software* que entran en la fabricación de aplicaciones informáticas. Pueden ser simples archivos, paquetes, bibliotecas cargadas dinámicamente, entre otros.

A continuación se presenta el diagrama de componentes para el sistema que se propone.

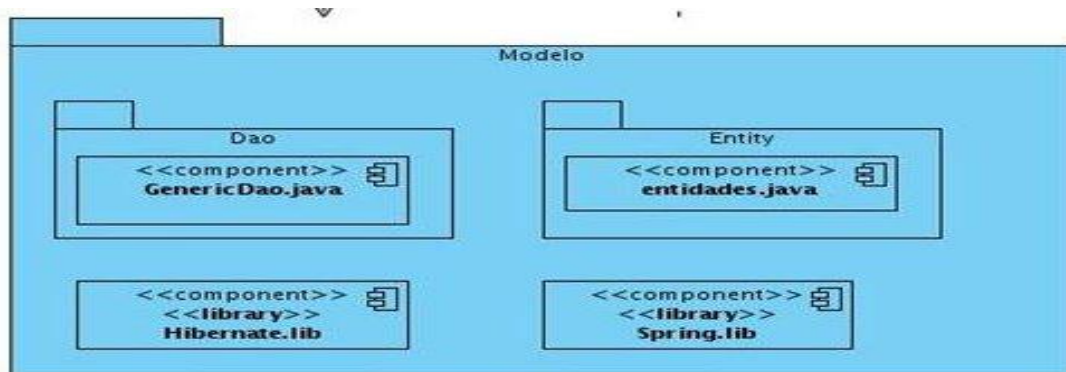


Figura 4 Diagrama de Componente.

Elementos que se encuentran dentro de los paquetes Dao y Entity.

Paquete Dao

Archivo	Descripción
GenericDao.java	Clase con todos los métodos genéricos para gestionar las entidades

Ejemplo de entidades que están dentro del paquete Entity.

Archivo	Descripción
Dadpeducacion.java	Describe todas las propiedades de los alumnos desaprobados por cada enseñanza.
Dalumnosatendidosespecialidad.java	Describe todas las propiedades de los alumnos atendidos por cada especialidad.
Dapaducacion.java	Describe todas las propiedades de

Capítulo 2: Diseño y Arquitectura de la base de datos.

	los alumnos aprobados por asignaturas en cada una de las enseñanzas.
Daprendizajealumnoeducacion.java	Describe todas las propiedades del aprendizaje del alumno.
Dcamyceeducacion.java	Describe todas las propiedades de la cobertura de atención médica y cobertura de enfermería.
Dcdeducacion.java	Describe todas las propiedades de los Criterios de distribución.
Dcegdgeducacion.java	Describe todas las propiedades de la continuidad de estudio de los graduados de duodécimo grado.
Dcengeducacion.java	Describe todas las propiedades de la continuidad de estudio de los alumnos noveno grado.
Dcentroseducacion.java	Describe todas las propiedades de los centros.
Dcentrospriorizadoseducacion.java	Describe todas las propiedades de los centros priorizados.
Dcesgreducacion.java	Describe todas las propiedades de la continuidad de estudio de los alumnos de sexto grado.
Dcomputadoraseducacion.java	Describe todas las propiedades de las computadoras.
Dcontinuidadestudioeducacion.java	Describe todas las propiedades de la continuidad de estudio.
Dcpeieducacion.java	Describe todas las propiedades de los centros priorizados e internos

Capítulo 2: Diseño y Arquitectura de la base de datos.

Dcpmeduccion.java	Describe todas las propiedades del cumplimiento del plan de mantenimiento.
Nmeses.java	Describe todas las propiedades de los meses.
Ddatosgeneraleseducacion.java	Describe todas las propiedades de los datos generales por cada escuela.
Dddlcpeducacion.java	Describe todas las propiedades del desglose de la cobertura por educaciones.
Dddppmeduccion.java	Describe todas las propiedades del desglose de presupuesto por meses.
Dddppmuneducacion.java	Describe todas las propiedades del desglose de presupuesto por municipio.
Ddiagnosticoorientacioneducacion.java	Describe todas las propiedades del diagnóstico de orientación.
Nmunicipios.java	Describe todas las propiedades de los municipios.

2.8 Diagrama de despliegue:

El Diagrama de Despliegue es un diagrama que se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes. Describen la topología del sistema, la estructura de los elementos de hardware y el software que ejecuta cada uno de ellos.

Los diagramas de despliegue representan a los nodos y sus relaciones. Los nodos son conectados por asociaciones de comunicación tales como enlaces de red, conexiones TCP/IP y microondas.

Capítulo 2: Diseño y Arquitectura de la base de datos.



Figura 5 Diagrama de Despliegue.

Cliente: El cliente se conectará por medio del protocolo WebSocket al Servidor jWebSocket para lo cual solo necesitará un navegador que soporte WebSocket sobre cualquier sistema operativo.

Servidor jWebSocket: El servidor de jWebSocket se conectará al servidor de base de datos de la Dirección a través del protocolo TCP/IP para persistir o consultar los datos.

2.9 Arquitectura de software

La arquitectura se refiere a la vista conceptual, la disciplina y arte encargada del estudio, análisis, organización, disposición y estructuración de la información en espacios de información, en este caso específicamente, Páginas Web.

En la actualidad todavía hay quienes no le dan importancia a este tema y desarrollan sitios sin una estructura significativa. En este proyecto en cambio, se preocupan por hacer un sitio que permita al usuario navegar fácil e intuitivamente por las distintas secciones, sin perder la orientación, con una presentación gráfica que sea visualmente atractiva y agradable, de fácil lectura, limpia y moderna. Por lo que se decidió para el desarrollo de esta aplicación de la arquitectura en capas.

La programación por capas es una arquitectura cliente-servidor en el que el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño; un ejemplo básico de esto consiste en separar la capa de datos de la capa de presentación al usuario. La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de que sobrevenga algún cambio, se ataca al nivel requerido sin tener que revisar entre código mezclado. Además, permite distribuir el trabajo de creación de una aplicación por niveles; de este

Capítulo 2: Diseño y Arquitectura de la base de datos.

modo, cada grupo de trabajo está totalmente abstraído del resto de niveles, de forma que basta con conocer la API que existe entre niveles.

En el presente trabajo de diploma se hace uso de una arquitectura en tres capas, la cual queda bien explicada en el siguiente epígrafe.

2.10 Implementación de la capa de Acceso

Trabajar con *software* orientado a objetos y bases de datos relacionales puede invertir mucho tiempo en los entornos actuales. Hibernate es una herramienta que realiza el *mapping* entre el mundo orientado a objetos de las aplicaciones y el mundo entidad-relación de las bases de datos en entornos Java y en cuanto a Spring, también ofrece varias opciones para la persistencia de datos.

Por las ventajosas prestaciones que proporciona Hibernate es el *frameworks* seleccionado para la creación de la base de datos y su capa de acceso a datos de la Dirección de Educación de la Administración Provincial de Artemisa.

Dicha dirección tendrá un módulo para la capa de acceso. El módulo estará conformado por una serie de ficheros como son: el `hibernate.cfg.xml`, los *Plain Old Java Objects* (POJOs), ficheros `hbm.xml`, entre otros.

Configuración de Hibernate

Los datos de configuración de la base de datos son registrados en un fichero `hibernate.properties` o `hibernate.cfg.xml` que debe estar en el *path* de la aplicación. En este fichero se indican los parámetros de conexión de la base de datos como la base de datos a conectar, usuario y *password*.

Plain Old Java Object (POJO)

Hibernate funciona asociando a cada tabla de la base de datos un Plain Old Java Object (POJO, a veces llamado Plain Ordinary Java Object). Un POJO es similar a una Java Bean, con propiedades accesibles mediante métodos `setter` y `getter`.

Fichero `hbm.xml`

Capítulo 2: Diseño y Arquitectura de la base de datos.

Para poder asociar el POJO a su tabla correspondiente en la base de datos, Hibernate usa los ficheros hbm.xml. En este tipo de fichero se declaran las propiedades del POJO y sus correspondientes nombres de columna en la base de datos, asociación de tipos de datos, referencias, relaciones con otras tablas, entre otras posibilidades.

Ejemplo de mapeo de clase usando archivos hbm.xml:

```
<? xml version="1.0"?>

<! DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<!-- Generated Apr 18, 2012 10:51:18 AM by Hibernate Tools 3.2.1.GA -->

< hibernate-mapping>

<class name="entity.Dmodeloeducacion" table="dmodeloeducacion" schema="public">

<id name="idModelo" type="int">

<columnname="id_modelo" />

<generator class="assigned" />

</id>

<property name="cursoEscolar" type="string">

<column name="curso_escolar" length="15" not-null="true" />

</property>

<property name="provincia" type="string">

<column name="provincia" length="10" not-null="true" />
```

Capítulo 2: Diseño y Arquitectura de la base de datos.

```
</property>

<property name="nombre" type="string">

<column name="nombre" length="50" not-null="true" />

</property>

</class>

</hibernate-mapping>
```

También otro recurso válido es mapear mediante el método denominado notación. Ejemplo de mapeo de clase usando notación:

```
@Entity

@Table(name="dmodeloeducacion"

, schema="mod_educacion"

)

Public class Dmodeloeducacion extends AbstractEntity {

private String cursoEscolar;

private String provincia;

private String nombre;

publicDmodeloeducacion() {}

publicDmodeloeducacion(String cursoEscolar, String provincia, String nombre) {

this.cursoEscolar = cursoEscolar;

this.provincia = provincia;
```

Capítulo 2: Diseño y Arquitectura de la base de datos.

```
this.nombre = nombre;

}

public Dmodeloeducacion(long id, String cursoEscolar, String provincia,

    String nombre) {

    this.id = id;

    this.cursoEscolar = cursoEscolar;

    this.provincia = provincia;

    this.nombre = nombre;

}

@Id

@SequenceGenerator(name="id_modelo",

sequenceName="mod_educacion.dmodeloeducacion_id_modelo_seq")

@GeneratedValue(generator="id_modelo")

@Column(name="id_modelo", unique=true, nullable=false)

@Override

public Long getId() {

return this.id;

}

@Column(name="curso_escolar", nullable=false, length=15)

public String getCursoEscolar() {
```

Capítulo 2: Diseño y Arquitectura de la base de datos.

```
returnthis.cursoEscolar;

}

public void setCursoEscolar(String cursoEscolar) {

    this.cursoEscolar = cursoEscolar;

}

@Column(name="provincia", nullable=false, length=10)

public String getProvincia() {

returnthis.provincia;

}

public void setProvincia(String provincia) {

this.provincia = provincia; }

@Column(name="nombre", nullable=false, length=50)

public String getNombre() {

returnthis.nombre;

}public void setNombre(String nombre) {

this.nombre = nombre;

}

}
```

Capítulo 2: Diseño y Arquitectura de la base de datos.

Conclusiones

En este capítulo se logró la realización del diseño de la BD para la Dirección de Educación, con el mismo se logra reducir la redundancia evitando la existencia de datos repetidos innecesariamente aumentando la integridad de los datos, evitando así que se creen anomalías en las consultas realizadas para extraer e insertar datos en el sistema. Se describieron los patrones de diseño a utilizar y el MER. Se analizó el modelo de la BD, para el almacenamiento de la información generada en la gestión de los procesos. Se presentaron las descripciones de algunas de las entidades. Además, se dieron a conocer algunas partes importantes de la programación de la capa de acceso a dato para lograr un correcto funcionamiento de la misma.

Capítulo 3. Adquisición y validación de los resultados del sistema.

Capítulo 3. Adquisición y validación de los resultados del sistema.

Introducción

En el presente capítulo se realiza la prueba y validación del diseño e implementación de la capa de acceso a datos de la Dirección de Educación de la Administración Provincial de Artemisa teniendo en cuenta ciertos aspectos como el duplicado e integridad de la información gestionada. El objetivo fundamental de este capítulo es validar el diseño realizado con aspectos teóricos y funcionales.

3.1 Duplicidad e integridad de los datos

Duplicado de datos

El duplicado de datos no es más que el almacenamiento repetido de los mismos. Esto puede significar un inconveniente al realizar modificaciones en los datos, siendo el elemento más frecuente de inconsistencias. Necesita además, mayor espacio de almacenamiento lo cual puede aumentar los costes de almacenamiento y acceso a los datos.

Integridad de los datos

La integridad de los datos hace alusión a la corrección y exactitud de la información gestionada. La integridad puede verse afectada de diferentes maneras al realizar modificaciones al contenido de los datos con operaciones de inserción, actualización y eliminación de información, a través de adiciones de datos inválidos y de modificaciones de datos existentes haciendo que tomen valores incorrectos o eliminándolos.

3.2 Ejecución de pruebas

Para garantizar el manejo y persistencia de la información de forma correcta se realizaron un conjunto de pruebas.

Capítulo 3. Adquisición y validación de los resultados del sistema.

3.2.1 Pruebas de Conexión

Se realizaron comprobaciones al archivo de configuración (hibernate-persistence.xml) que necesita los *frameworks* Spring e *Hibernate* para la conexión con la base de datos.

La sentencia del archivo de configuración es la siguiente

```
<!--<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">
-->
<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"
xmlns:jee="http://www.springframework.org/schema/jee"
xmlns:lang="http://www.springframework.org/schema/lang"
xmlns:p="http://www.springframework.org/schema/p"
xmlns:tx="http://www.springframework.org/schema/tx"
xmlns:util="http://www.springframework.org/schema/util"
xmlns:aop="http://www.springframework.org/schema/aop"
xsi:schemaLocation="http://www.springframework.org/schema/lang http://www.springframework.org/schema/lang/spring-
lang.xsd
http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee.xsd
http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-3.0.xsd
http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-
beans-3.0.xsd
http://www.springframework.org/schema/util http://www.springframework.org/schema/util/spring-util.xsd
http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx.xsd
http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-
context.xsd">
<bean id="myDataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
<property name="driverClassName" value="org.postgresql.Driver"/>
<property name="url" value="jdbc:postgresql://10.208.1.250:5432/sigob"/>
<property name="username" value="user"/>
<property name="password" value="user"/>
</bean>
```

Capítulo 3. Adquisición y validación de los resultados del sistema.

```
<bean id="sessionFactory"
      class="org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean">
  <property name="dataSource" ref="myDataSource" />
  <!--<property name="annotatedClasses">
    <list>
      <value>hab.uci.cu.model.entity.Project</value>
      <value>hab.uci.cu.model.entity.Country</value>
    </list>
  </property-->
  <property name="packagesToScan">
    <list>
      <value>entity</value>
    </list>
  </property>
  <property name="hibernateProperties">
    <props>
      <prop key="hibernate.dialect">org.hibernate.dialect.PostgreSQLDialect</prop>
      <prop key="hibernate.show_sql">>true</prop>
    </props>
  </property>
</bean>

<!--<tx:annotation-driven />-->
<bean id="transactionManager"
      class="org.springframework.orm.hibernate3.HibernateTransactionManager">
  <property name="sessionFactory" ref="sessionFactory" />
</bean>

<bean id="genericDAO" class="dao.GenericDao">
  <property name="sessionFactory" ref="sessionFactory"/>
</bean>

<bean id="projectService" class="testdb.TestDB">
  <property name="dao" ref="GenericDao"/>
</bean>

</beans>
```

3.2.2 Pruebas funcionales realizadas y resultados

Se pusieron en práctica pruebas para comprobar la capacidad funcional de los métodos declarados en la capa de acceso a datos verificando de esta forma el cumplimiento de los requisitos funcionales. Para la comprobación se utilizó una clase principal en java, óptima para este propósito. En la misma fueron probadas

Capítulo 3. Adquisición y validación de los resultados del sistema.

las diferentes operaciones de los métodos de la capa de acceso como: inserción, modificación, búsqueda y eliminación de registros.

```
public class EducacionFinal {
private static GenericDao dao1;
/**
 * @paramargs the command line arguments
 */
public static void main(String[] args) throws ParseException {
// TODO code application logic here
Resource resource = new
FileSystemResource("/home/odelalisy/NetBeansProjects/EducacionFinal/hibernate-
persistence.xml");
XmlBeanFactory bean = new XmlBeanFactory(resource);
dao1=(GenericDao) bean.getBean("genericDAO");
```

Este constituye el método principal de una clase java en consola. En su cuerpo se inicializa una instancia del GenericDao para acceder a cada una de sus funcionalidades.

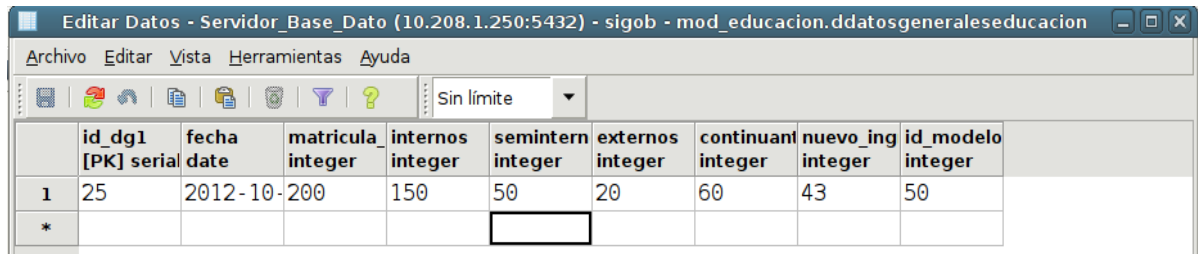
Los métodos de prueba fueron realizados para cada entidad de la base de datos. Algunas de las pruebas realizadas a la capa de acceso correspondiente al módulo de la Dirección de Educación de la Administración Provincial de Artemisa fueron las siguientes:

En este caso la prueba se realizó en la entidad Ddatosgeneraleseducacion.

Insertar DatosGenerales

```
List<Dmodeloeducacion>modelo=dao1.findByParamAndValue(Dmodeloeducacion.class,
"cursoEscolar", "2011-2012");
Ddatosgeneraleseducaciondge=new Ddatosgeneraleseducacion(modelo.get(0), fecha, 200,
150, 50, 20, 60, 43);
dao1.save(dge);
```

Capítulo 3. Adquisición y validación de los resultados del sistema.



	id_dg1 [PK] serial	fecha date	matricula integer	internos integer	semintern integer	externos integer	continuant integer	nuevo_ing integer	id_modelo integer
1	25	2012-10	200	150	50	20	60	43	50
*									

Figura 6 Insertar datos generales.

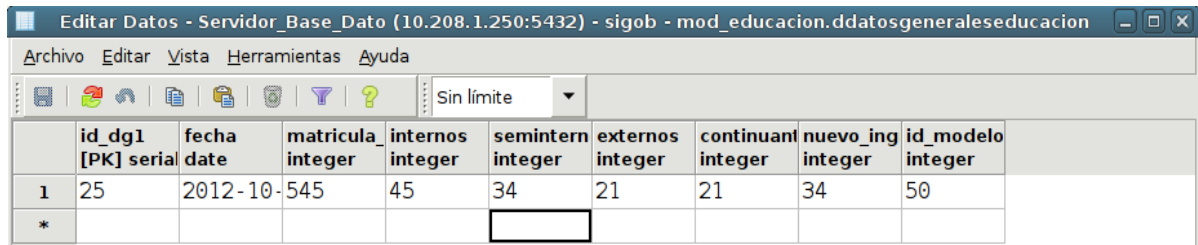
Modificar Ddatosgeneraleseducacion

```
List<Dmodeloeducacion>modelo=dao1.findByParamAndValue(Dmodeloeducacion.class,  
"cursoEscolar", "2011-2012");
```

```
List<Ddatosgeneraleseducacion>ldge=dao1.findByParamAndValue(Ddatosgeneraleseduca  
cion.class, "dmodeloeducacion", modelo.get(0));
```

```
Ddatosgeneraleseducaciondge=new Ddatosgeneraleseducacion(ldge.get(0).getId(),  
modelo.get(0), fecha, 545, 45, 34, 21, 21, 34);
```

```
dao1.update(dge);
```



	id_dg1 [PK] serial	fecha date	matricula integer	internos integer	semintern integer	externos integer	continuant integer	nuevo_ing integer	id_modelo integer
1	25	2012-10	545	45	34	21	21	34	50
*									

Figura 7 Modificar datos generales.

Eliminar Ddatosgeneraleseducacion

```
List<Dmodeloeducacion>modelo=dao1.findByParamAndValue(Dmodeloeducacion.class,  
"cursoEscolar", "2011-2012");
```

```
List<Ddatosgeneraleseducacion>ldge=dao1.findByParamAndValue(Ddatosgeneraleseduca  
cion.class, "dmodeloeducacion", modelo.get(0));
```

```
dao1.delete(ldge.get(0));
```

Capítulo 3. Adquisición y validación de los resultados del sistema.

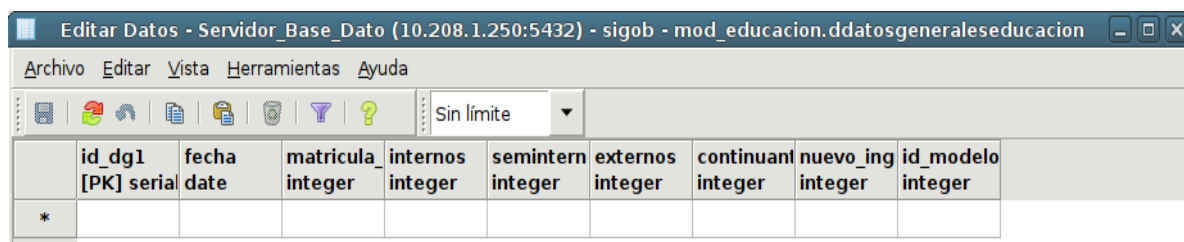


Figura 8 Eliminar datos generales.

3.3 Resultados y Funcionalidades obtenidas

A partir de la Propuesta de Solución queda disponible la Base de datos con su capa de acceso en su versión 1.0, la cual tiene como funcionalidades principales insertar, buscar, modificar y eliminar cualquier tipo de información referente a dicha Dirección. Esta le proporciona al usuario integridad, confiabilidad y centralidad en cuanto al manejo de la información.

3.4 Aporte Social y Económico

Actualmente en la Dirección de Educación no se encuentra informatizado ninguno de los procesos que allí se desarrollan, realizándose el trabajo de forma manual implicando un mayor tiempo y esfuerzo por parte de los especialistas encargados de procesar una gran cantidad de información referente a todos los procesos educacionales que tienen lugar en la provincia. El análisis de dicha información generada en la Dirección podría estar propenso a futuros errores impidiendo tener valores confiables de toda la labor educativa realizada por parte del personal docente y de la real situación educacional que poseen los estudiantes de las diferentes enseñanzas. A través de dicha base de datos toda la información que se maneja en dicha dirección estará centralizada permitiendo un correcto almacenamiento de la información. Esta base de datos tiene la ventaja económica que es realizada en Cuba, es decir no usa un *software* propietario que haya que pagar para su uso, ni es necesario invertir gran cantidad de dinero contratando una serie de desarrolladores para la realización de la misma. Esta puede ser implantada en cualquier provincia o municipio que estén necesitando de ella sin necesidad de pagar para ser utilizada.

Capítulo 3. Adquisición y validación de los resultados del sistema.

Conclusiones

En este capítulo se presentaron los aspectos necesarios para realizar la validación práctica de la solución propuesta. Con la validación de la capa de acceso se pusieron a prueba una serie de elementos importantes como el control ante duplicados de información y el manejo de datos. Por medio de las pruebas realizadas se pudo tener conocimiento de cómo respondería la capa de acceso a datos ante determinadas situaciones. El comportamiento fue normal y esperado obteniendo resultados satisfactorios.

Conclusiones Generales

Conclusiones Generales

Durante el desarrollo de este trabajo se expuso la necesidad de desarrollar una base de datos para la Dirección de Educación de la Administración Provincial (AP) de Artemisa que de soporte a la gestión de los procesos que allí se desarrollan.

Luego del estudio realizado de los procesos de la Dirección de Educación que tienen lugar actualmente en la Administración Provincial de Artemisa en aras de lograr informatizar dicho sector, así como el cumplimiento del objetivo y tareas trazados, se arribó a las siguientes conclusiones:

- ✓ Se realizó un estudio concreto de las tendencias y tecnologías actuales, permitiendo seleccionar las herramientas adecuadas para el desarrollo de la solución propuesta.
- ✓ Fueron modelados los diagramas relacionales (teniendo en cuenta la representación de la estructura lógica y física) de la base de datos, resaltándose la descripción de cada entidad y sus relaciones.
- ✓ Se desarrolló la propuesta de construir una base de datos usando PostgreSQL como SGBD, para implementar la BD.
- ✓ Se empleó la tecnología Hibernate-Spring para el acceso a datos, con el mapeo de las tablas de la BD utilizando el patrón de diseño DAO.
- ✓ Se tuvo en cuenta la integridad de la información, las validaciones de los campos de cada tabla, además de implementarse la capa de acceso a datos, verificándose si la información es correctamente almacenada y todo con vista a las funcionalidades principales de la base de datos. Se integró en su uso la plataforma de Java, con ésta el framework Jwebsocket y la tecnología de hibernate-spring para desarrollar su conectividad con la BD.

Luego de estos procesos de trabajo realizados se considera que se han cumplido con los objetivos planteados y se puede concluir que la Base de Datos para la Dirección de Educación de la Administración Provincial de Artemisa dará solución a la situación problemática planteada y su implantación supondrá una mejora de consideración en la gestión de los procesos de dicha Dirección.

Recomendaciones

Recomendaciones

- ✓ Que finalmente sea implantada y puesta en funcionamiento la base de datos producto de esta investigación.
- ✓ Exigir y velar por el estricto cumplimiento del proceso de mantenimiento y copias de seguridad periódicas, logrando así que se mantenga la fiabilidad y funcionamiento óptimo de la base de datos.
- ✓ Continuar con la investigación para garantizar nuevas mejoras en futuras versiones de la base de datos del sistema propuesto.

Referencias Bibliográficas.

Referencias Bibliográficas

1. [En línea] [Citado el: 20 de 10 de 2011.] <http://definicion.de/gestion>.
2. Portal Educativo de Argentina. [En línea] <http://www.educ.ar/educar/gestion-institucional-conceptos-introductorios.html> .
3. [En línea] [Citado el: 13 de 11 de 2011.] <http://definicion.de/informacion/>.
4. Chiavenato, Idalberto. *Introducción a la Teoría General de la Administración*. s.l. : McGraw-Gill Iteramericana, 2006.
5. Pinto, Maria. BÚSQUEDA Y RECUPERACIÓN DE INFORMACIÓN. *BÚSQUEDA Y RECUPERACIÓN DE INFORMACIÓN*. [En línea] 13 de 04 de 2011. [Citado el: 2 de 01 de 2012.] [Disponible en:http://www.mariapinto.es/e-coms/recu_infor.htm.
www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=PersistenciaJava.
6. Lagares Abogados, Canalejas, 29 Las Palmas de Gran Canaria.[Disponible en: <http://definicion.de/integridad/>]
7. Escalona, Iván. Factores Universales para determinar la Confiabilidad.[Disponible en: <http://www.monografias.com/trabajos16/confiabilidad/confiabilidad.shtml>]
8. Arribas Urrutia, Amaia. Universidad del País Vasco. ¿Centralizar o descentralizar los sistemas de información en la empresa?. [Disponible en: <http://grupo.us.es/grehcco/ambitos03-04/03urrutia.pdf>]
9. I, Rafea Mendez. Metodologías de desarrollo. *Metodologías de desarrollo*. [En línea] 2011. [Citado el: 26 de 09 de 2011.]
<http://www.um.es/docencia/barzana/IAGP/IAGP2-Metodologias-de-desarrollo.html>.
10. Sanchez, María A Mendoza. *Metodologías De Desarrollo De Software*. 2011.
11. Penadés, Patricio LetelierTorres y Carmen Maria. *Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP)*. 2011.
12. Javeriana, Universidad. Historia de las bases de datos en Ciencia de la Informacion. *Historia de las bases de datos en Ciencia de la Informacion*. [En línea] 2006. [Citado el: 2 de 11 de 2011.]
http://recursostic.javeriana.edu.co/wiki/index.php/Historia_de_las_bases_de_datos_en_Ciencia_de_la_Informaci%C3%B3n.

Referencias Bibliográficas.

13. Álvarez, Sara. Arquitectura de las bases de datos. *Arquitectura de las bases de datos*. [En línea] 2007. [Citado el: 1 de 12 de 2011.]
www.desarrolloweb.com/articulos/arquitectura-base-de-datos.html.
14. SEVILLA, Universidad. 2008. Diseño de Bases de Datos. Departamento de Lenguajes y Sistemas Informáticos. 16p
15. Mexico, Instituto Politécnico. Modelos de Datos. *Modelos de Datos*. [En línea] 2009. [Citado el: 12 de 11 de 2011.]
http://sistemas.itlp.edu.mx/tutoriales/basedat1/tema1_4.htm.
16. R, Karen M Zequera. *Rol de Diseñador de Bases de Datos*. Universidad de las Ciencias Informáticas, Cuba : Plataforma de Televisión Informativa, 2009.
17. Álvarez, Sara. Modelos de bases de datos. *Modelos de bases de datos*. [En línea] 2007. [Citado el: 3 de 10 de 2011.]
<http://www.desarrolloweb.com/articulos/modelos-base-datos.html>.
18. Sanchez, Jorge. *Sistemas Gestores de Base de Datos*. California : Palo Alto, 2009. 182p.
19. HIBERNATE. Framework Hibernate. *HIBERNATE. Framework Hibernate*. [En línea] 2011. [Citado el: 24 de 11 de 2011.] <http://www.hibernate.org/>.
20. García, Yanly Suárez. *Diseño de la base de datos para el Grupo de Calidad de la facultad 9*. Universidad de las Ciencias Informáticas, Cuba : s.n., 2010. 74p.
21. Peñalver, Gladis. □MA-GMPR-UR2 Metodología ágil para proyectos de software libre. □MA-GMPR-UR2 Metodología ágil para proyectos de software libre. [En línea] 2008. [Citado el: 10 de 02 de 2012.]
http://bibliodoc.uci.cu/TD/TD_1309_08.pdf.
22. EVA. (2010). Recuperado el 28 de febrero de 2011, de Conferencia de normalización:http://eva.uci.cu/file.php/624/2._Clases/Semana_5/2da_frecuencia/MApoyo/C4_Normaliz.pdf.
23. Rojas, Juan Carlos Olivares. *Patrones de Diseño*. Cuba : s.n., 2010.
24. Rodríguez, Maiby Pérez. *Módulo para la elaboración del Anteproyecto de Planificación del Sistema*. Cuba : CEDRUX, 2009.
25. Tedeschi, Nicolás. ¿Qué es un Patrón de Diseño? ¿Qué es un Patrón de Diseño? [En línea] 2012. [Citado el: 20 de 01 de 2012.]
<http://msdn.microsoft.com/es-es/library/bb972240.aspx>.

Referencias Bibliográficas.

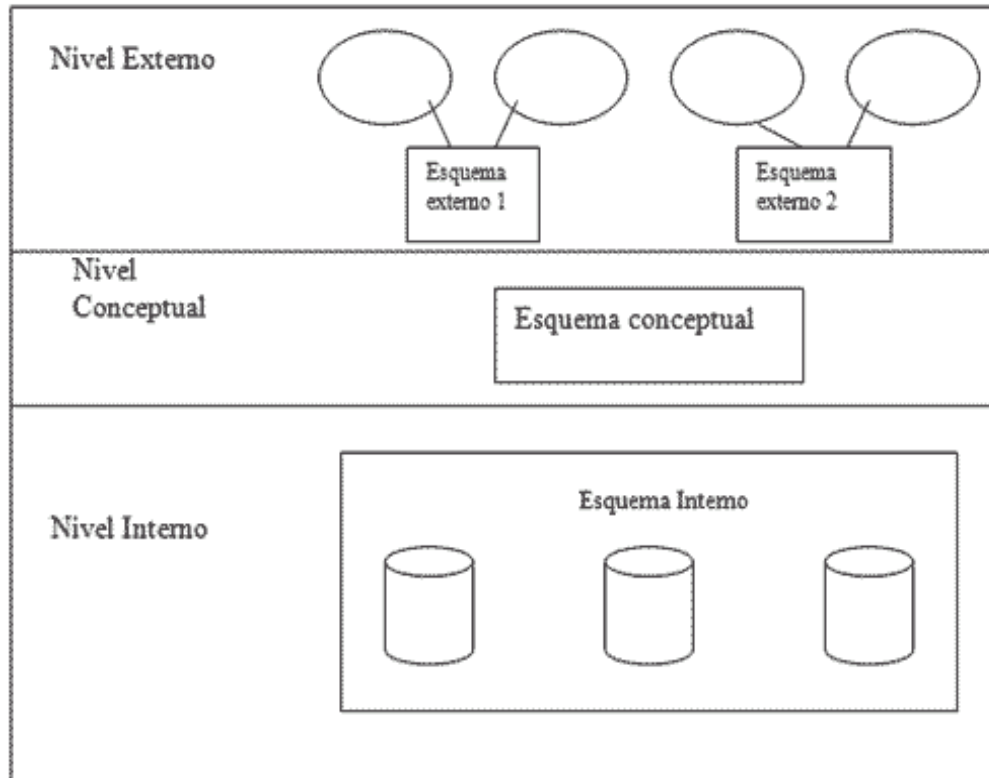
26. Tecnología de la Información y la Comunicación. Base de datos. *Tecnología de la Información y la Comunicación. Base de datos*. [En línea] 2007. [Citado el: 22 de 01 de 2012.] www.belgrano.esc.edu.ar/./carpeta_de_access_introduccion.pdf.
27. Hernández, Yenisleydis Rodríguez Martínez y Madisleidy Casanova. *Diseño e implementación de una herramienta webmétrica libre para el análisis personalizado de sitios webs para GEWEB*. Cuba : s.n., 2011. 78.

Bibliografía

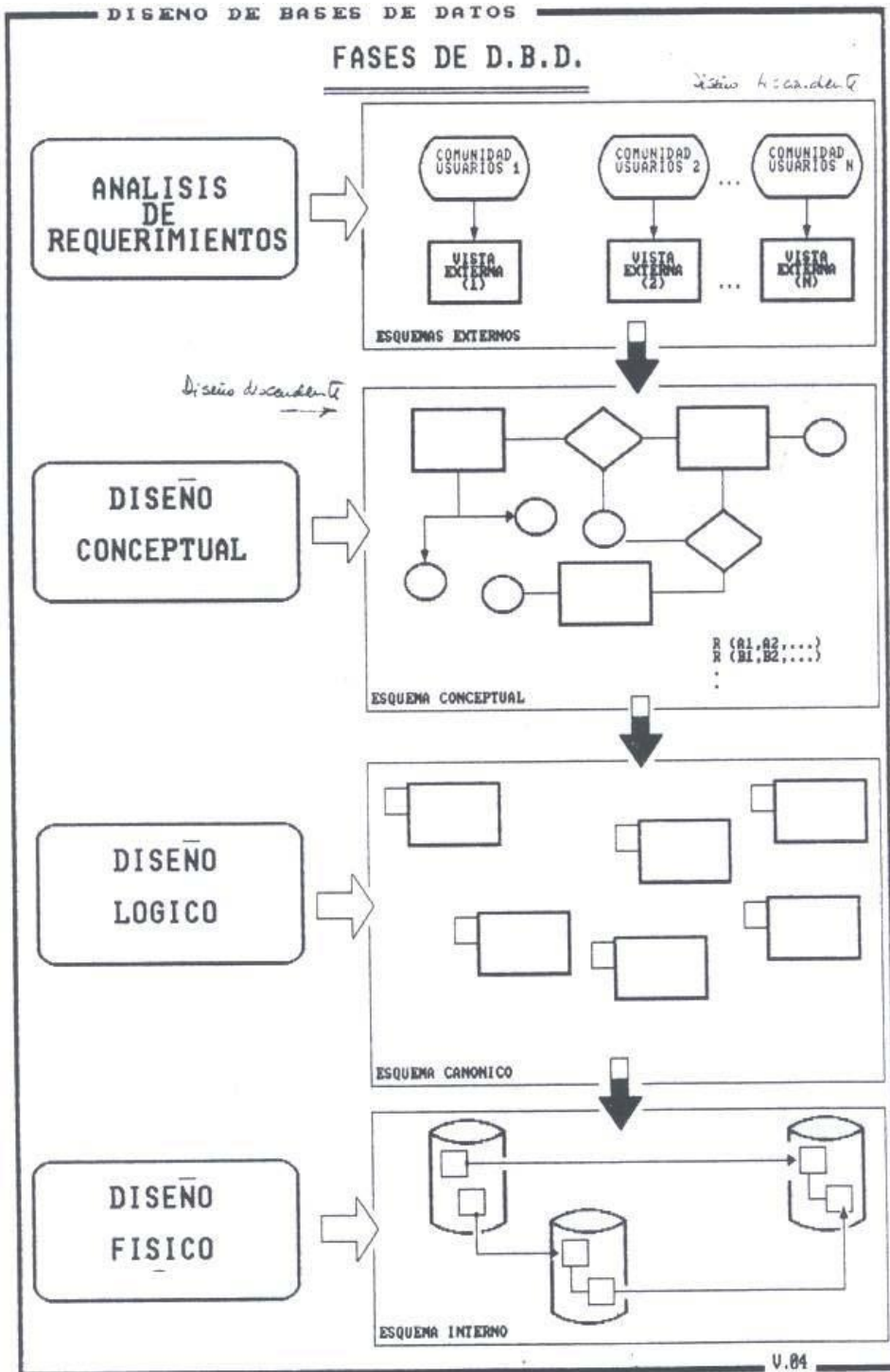
Bibliografía

1. DARA, G. *Información detallada módulos SIGA* Última actualización: 2009. [Disponible en: <http://www.dara.es/siga/sigainf.htm>].
2. ORQUÍN, A. F.; BARRIOS, Y. *Sistema para la recuperación de información docente*. Editado por:. Publicado el: 2006 de 2006, última actualización: 2006. [Consultado el: febrero 2, 2012]. [Disponible en: http://informaticahabana.com/evento_virtual/files/EDU089.doc].
3. Alfredo. 2011. Illinois. [En línea] 2011. <http://st-www.cs.illinois.edu/users/smarch/st-docs/mvc.html>.
4. Cisneros, Jonathan. 2007. Cisneros. [En línea] 24 de 09 de 2007. <http://cisneros.wordpress.com/2007/09/24/creando-una-capa-de-acceso-a-datos>.

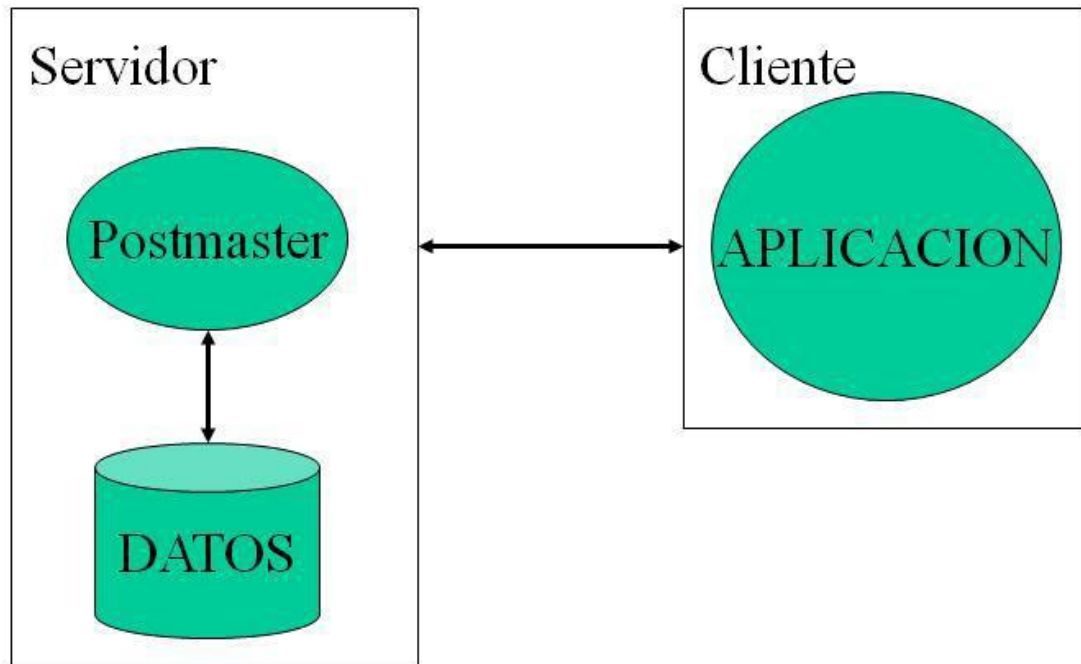
Anexos



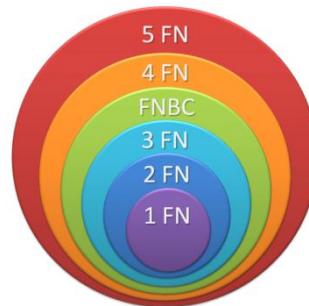
Anexo#1: Arquitectura de las Bases de Datos.



Anexo#2: Fases del diseño de una Base de Datos.



Anexo#3. Arquitectura de PostgreSQL.



Ver Anexo#4: Diagrama de inclusión de todas las formas normales.

Glosario

Atributos: son características o propiedades asociadas a la entidad que toman valor en una instancia particular. Ejemplo: nombre, cédula, teléfono.

Base de Datos: Desde el punto de vista informático, la base de datos es un sistema formado por un conjunto de datos almacenados en discos que permiten el acceso directo a ellos y un conjunto de programas que manipulen ese conjunto de datos.

CASE: (Computer-Aided Software Engineering, Ingeniería de Software Asistida por Ordenador). Diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero.

Clave principal o primaria: se le llama al atributo o conjunto mínimo de atributos (uno o más campos) que permiten identificar en forma única cada instancia de la entidad, es decir, a cada registro de la tabla.

Clave foránea: (también llamada externa o secundaria) es un atributo que es clave primaria en otra entidad con la cual se relaciona.

Cardinalidad: Describe la dimensión cuantitativa en la relación entre un par de entidades.

DAO (Data Access Object): Objeto de Acceso a Datos.

Entidad: es cualquier objeto, real o abstracto, que existe en un contexto determinado o puede llegar a existir y del cual se puede guardar información.

Framework: Un *frameworks* denota un conjunto de objetos que definen un diseño abstracto para solucionar un conjunto de problemas relacionados. Puede incluir programas, bibliotecas de objetos o lenguaje interpretado, por lo que su uso facilita la elaboración de sistemas informáticos.

Glosario

Gestor de base de datos: Es un tipo de *software* específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta.

Hardware: corresponde a todas las partes físicas y tangibles de una computadora.

Modelo entidad-relación: Es una técnica para el modelado de datos utilizando diagramas de entidad-relación.

Normalización: Proceso de reducción sobre una estructura de datos que procura aumentar la integridad, disminuir la redundancia y las dependencias funcionales de esa estructura.

ORM: Acrónimo de Object-Relational Mapping (Mapeo Objeto-Relacional).

Framework que utiliza técnicas de programación para convertir datos a objetos y viceversa, permitiendo el trabajo con datos persistentes como si formaran parte de una Base de Datos orientada a objetos.

Software: se refiere al equipamiento lógico o soporte lógico de una computadora digital, y comprende el conjunto de los componentes lógicos necesarios para hacer posible la realización de tareas específicas.