Facultad Regional Mártires de Artemisa

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Título:

Base de Datos y Capa de Acceso a Datos para los Módulos del CITMA y Planificación Física de la Administración Provincial de Artemisa.

Autor:

Luis Ernesto Miralles Díaz

Tutor(a):

Ing. Raisa Ortega Báez

Co-Tutor(a):

Ing. Dayana Cánova Ramírez

Artemisa, Junio 2012

"Nunca consideres el estudio como una obligación, sino como una oportunidad para penetrar en el bello y maravilloso mundo del saber ".

ALBERT EINSTEIN

Declaración de Autoría

Declaro que soy el único autor de este trabajo y autorizo a la “Facultad Regional Mártires de Artemisa” de la Universidad de las Ciencias Informáticas; así como a dicho centro para que hagan el uso que estimen pertinente con este trabajo.

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

_____ _____

Autor: Luis Ernesto Miralles Díaz Tutora: Ing. Raisa Ortega Báez

_____

Co-Tutora: Ing. Dayana Cánova Ramírez

AGRADECIMIENTOS

A mis padres por todo el apoyo y el cariño que me han dado todo este tiempo.

A mis hermanos por su ayuda incondicional.

A todos mis compañeros del antiguo grupo 4 por su amistad.

A Sonia por su amor, su cariño y todo el apoyo que me ha dado durante los 5 años que hemos pasado juntos.

A Diana por su ayuda y buenos consejos.

A mis compañeros de cuarto por los gratos momentos que hemos compartido juntos.

A Aylen, Amed y todos los que me brindaron sus consejos y críticas constructivas.

A mi tutora y co-tutora por sus consejos y ayuda en la investigación.

A todos los profesores que han contribuido a mi formación.

En fin a todos aquellos que de una forma u otra me han brindado su amistad y me han dado su apoyo.

A todos muchas gracias.

DEDICATORIA

De manera especial a mis padres por haberme apoyado incondicionalmente y darme todo el cariño.

A mis hermanos por ayudarme en todo momento.

Resumen

Este trabajo surge como una necesidad de la Administración Provincial de Artemisa de automatizar los procesos que se llevan a cabo en las direcciones que la componen. Esta investigación tiene como objetivo desarrollar la capa de acceso a datos y la base de datos para los Módulos del CITMA y Planificación Física de la Administración Provincial de Artemisa.

En el presente documento se describe detalladamente el diseño de una base de datos que garantiza la manipulación de la información en estas direcciones. Se detalla también la implementación de la capa de acceso a datos para lograr una eficaz conexión a la bases de datos. Con el fin de cumplimentar los objetivos propuestos fue realizada una investigación acerca de las tecnologías existentes relacionadas con el acceso a los datos en base de datos a nivel mundial. Finalmente, la solución fue sometida a un proceso de validación teórica y funcional.

Para el desarrollo de la solución se utilizaron las capacidades de los Frameworks Hibernate y Spring y el Entorno de Desarrollo Integrado NetBeans 7.0.1.

Palabras claves: Base de Datos; Capa de Acceso a Datos; Frameworks.

ÍNDICE

Introducción 1
Capítulo 1: Fundamentación teórica. 8
1.1 Gestión de información de una base de datos. 8
1.2 Importancia de la Capa de Acceso a Datos. 10
1.3 Conceptos y características de Base de Datos. 11
1.4 Clasificación de las bases de datos. 12
1.5 Fases del diseño de una base de datos. 13
1.6 Modelos de Bases de Datos. 13
1.7 Sistemas de Gestión de Bases de Datos. 16
1.8 Metodologías de Desarrollo de Software. 21
1.9 Herramientas Utilizadas. 23
1.10 Frameworks de desarrollo a utilizar. 24
1.11 Estado del Arte. 25
Capítulo 2: Diseño y arquitectura de la capa de acceso a datos. 29
2.1 Diseño de la BD. 29
2.2 Clases Persistentes. 34
2.3 Descripción de la arquitectura. 38
2.4 Estructura y elementos de paquete. 39
Capítulo 3: Implementación y validación de la capa de acceso a datos. 44
3.1 Implementación DAO Genérico. 44
3.2 Mapeo Objeto Relacional (ORM). 47
3.3 Validación teórica del diseño. 50
3.4 Pruebas a la capa de acceso a datos. 53
Conclusiones Generales. 58
Recomendaciones 59
Glosario de Términos. 65
Anexos. 66

ÍNDICE DE IMÁGENES

Imagen 1. Estructura de la aplicación. 39
Imagen 2. Método save. 44
Imagen 3. Método saveOrUpdate. 45
Imagen 4. Método find. 45
Imagen 5. Método findByPK. 45
Imagen 6. Método findByParamAndValue. 45
Imagen 7. Metodo findByParamsAndValues. 46
Imagen 8. Método update. 46
Imagen 9. Método delete. 46
Imagen 10. Método deleteAll. 46
Imagen 11. Archivo de configuración de Hibernate. 48
Imagen 12. Archivo de ingeniería inversa de Hibernate. 48
Imagen 13. Estructura general de los POJOS en Hibernate. 49

ÍNDICE DE TABLAS

Tabla 1. Descripción de la tabla módulo. 32
Tabla 2. Descripción de la tabla de las Empresas Inscriptas. 32
Tabla 3. Descripción de la tabla de las Áreas Protegidas. 32
Tabla 4. Descripción de la tabla de los Organismos Infractores. 33
Tabla 5. Descripción de la tabla de Denominación. 33
Tabla 6. Descripción de la tabla de Avance de Ejecución de Inversiones. 34
Tabla 7. Descripción de la clase Dmodulocitma. 35
Tabla 8. Descripción de la clase Dempinscriptascitma. 36
Tabla 9. Descripción de la clase Dareasprotegidascitma. 36
Tabla 10. Descripción de la clase Dorganismoinfractpfisica. 37
Tabla 11. Descripción de la clase Ddenominacionpfisica. 38
Tabla 12. Descripción de la clase Davanceejecinverspfisica. 38
Tabla 13. Descripción del componente dao. 40
Tabla 14. Descripción del componente citma. 42
Tabla 15. Descripción del componente phisica_planing. 43
Tabla 16. Prueba al Método save. 54
Tabla 17. Prueba al Método saveOrUpdate. 54
Tabla 18. Prueba al Método find. 54
Tabla 19. Prueba al Método findByPK. 55
Tabla 20. Prueba al Método findByParamAndValue. 55
Tabla 21. Prueba al Método findByParamsAndValues. 55
Tabla 22. Prueba al Método update. 56
Tabla 23. Prueba al Método delete. 56
Tabla 24. Prueba al Método deleteAll. 56

Introducción

La información constituye sin lugar a dudas un recurso vital para el desarrollo de las organizaciones. Hoy en día el manejo de la información es lo que determina el éxito o el fracaso de los procesos que se llevan a cabo en una entidad, de manera que establecen en gran medida que la organización alcance cierto nivel competitivo en el mercado, es por esto que desde hace algún tiempo las empresas han reconocido la importancia que tiene el buen manejo de la información.

El almacenamiento, la gestión y el análisis de los datos por parte de las empresas u organizaciones se ha hecho cada vez más dependiente de la tecnología, debido a los crecientes volúmenes de datos que se manejan en las mismas. Para las empresas se hace necesario controlar de manera eficiente los grandes volúmenes de información generada por los diferentes procesos que se llevan a cabo, lo que posibilita tener mayor control de los recursos accediendo a los datos de forma confiable y precisa.

La necesidad de almacenar grandes cantidades de datos para luego ser consultados fue una de las principales causas que dio origen al desarrollo de las bases de datos. Estás son el soporte principal para las instituciones en sus procesos habituales de gestión, apoyando así a la toma de decisiones y contribuyendo a brindar servicios más eficientes. El uso de bases de datos influye positivamente en el desarrollo de una entidad, incluso en el progreso de un país.

El siglo XXI se ha caracterizado por el desarrollo acelerado de las Tecnologías de la Información y las Comunicaciones (TIC), estas constituyen un apoyo fundamental para la gestión del conocimiento, ayudando a la consolidación de la sociedad de la información. El uso de las TIC se hace cada vez más imprescindible en la sociedad, pues ofrece un nuevo espacio de innovación en diferentes ámbitos como la industria, los servicios, la salud, la administración, el comercio, la educación entre otros.

Dentro del entorno empresarial, la incorporación de las TIC se hace una necesidad. Su explotación beneficia a las organizaciones en diferentes aspectos como la digitalización de los datos, proveyendo a las empresas de múltiples mecanismos que mejoran el almacenamiento y análisis de la información que cada una de ellas manejan. Con la evolución de las TIC se han creado disímiles sistemas y herramientas que han dando un gran aporte a las distintas actividades humanas donde se concretan en una serie de funciones que facilitan la realización de trabajos, donde se requiere de una cierta información para realizarlo y un determinado proceso de gestión de datos.

Los Sistemas Gestores de Base de Datos (SGBD) no han quedado exentos a los avances de las TIC, convirtiéndose estos en herramientas esenciales para lograr la gestión de la información. Estos sistemas permiten almacenar, organizar y efectuar consultas de datos, en breves períodos de tiempo y de forma eficaz. Proporcionando nuevos métodos para analizar los datos, encontrar correlaciones y dependencias entre ellos.

La mayoría de las organizaciones en el mundo se han centrado en la implantación de nuevos sistemas que automaticen los procesos de negocio, buscando incrementar su productividad y ganar ventaja competitiva. Las aplicaciones de almacenamiento de datos más frecuentes son para la gestión de empresas e instituciones públicas destacándose en este sentido las organizaciones dedicadas a las finanzas, las ventas, la producción, los recursos humanos, las telecomunicaciones. También son ampliamente utilizadas en entornos científicos con el objetivo de recopilar la información experimental.

La aplicación y el desarrollo de las bases de datos en Cuba se han ido incrementando considerablemente en los últimos años, a partir de la utilización de las TIC en todas las esferas de la sociedad cubana. Por las posibilidades que brinda al país el uso del software libre se ha visto una progresiva migración hacia los SGBD de código abierto como el Postgres.

Cuba a pesar de ser un país subdesarrollado, con pocos recursos económicos, estar sometida a un férreo bloqueo económico y no tener acceso a los principales avances de la ciencia y la técnica, con los que se cuentan actualmente a nivel mundial, no está exenta del desarrollo tecnológico.

El Estado cubano ha llevado a cabo disímiles proyectos con el objetivo de informatizar y automatizar los principales sectores de la sociedad en busca del perfeccionamiento para lograr el éxito de las actividades en las entidades, entre estos se destaca la creación de la Universidad de las Ciencias Informáticas (UCI) en el año 2002.

La UCI fue concebida como pilar estratégico para apoyar el proceso de informatización de la sociedad que se está llevando a cabo en el país. Esta universidad implementa un nuevo modelo de formación donde se vincula el estudio con la producción e investigación favoreciendo de esta forma la producción de software para la industria nacional y la exportación dándole grandes aportes a la economía del país.

La UCI posee convenios con diferentes entidades, logrando avances significativos en la informatización de las mismas. Ejemplo de ello es el convenio existente entre la Facultad Regional Mártires de Artemisa y la Administración Provincial de Artemisa, surgido a raíz de la nueva división política-administrativa que se llevó a cabo en el país a inicio del 2011 con el fin de contribuir a la toma de decisiones como factor de éxito para la Provincia de Artemisa.

La Administración Provincial de Artemisa cuenta con 32 direcciones provinciales entre las cuales se encuentran las direcciones del CITMA y Planificación Física. Esta última maneja de forma general todas las gestiones que se realizan a nivel provincial en cuanto al control mensual de la cantidad y tipo de ilegalidades detectadas en cada uno de los municipios, ya sean infractores particulares u organismos infractores, y el control mensual del avance de ejecución en las Inversiones Nominales.

La dirección del CITMA tramita la información referente a todas las gestiones que se realizan a nivel provincial respecto al medio ambiente, los eventos de innovación e investigación que realizan las entidades de la región, ya sean pertenecientes a las ramas de la Educación, la Agricultura, el control de los puertos y bahías ubicados dentro de la provincia y las penalidades y legislaciones del CITMA que corrigen y controlan el funcionamiento de las mismas.

Actualmente en estas direcciones el proceso de almacenamiento de la información se realiza de forma manual, en formato duro o digital, lo que trae consigo demora en la tramitación de los datos así como pérdida, duplicado y falta de seguridad de los mismos, los reportes estadísticos que se generan son poco fiables por lo difícil que resulta recopilar un gran volumen de información al transcurrir largos períodos de tiempo, la consulta de la información resulta ser un proceso engorroso pues requiere de un costo elevado de tiempo y esfuerzo.

Todos estos conflictos traen consigo deficiencias para la generación de reportes inmediatos, la búsqueda de información relacionada con la dirección así como su entrega al Presidente de la Administración Provincial, en el tiempo determinado y con la calidad requerida. Teniendo en cuenta las dificultades descritas anteriormente se define como **problema a resolver** ¿Cómo mejorar la persistencia y recuperación de la información en las direcciones del CITMA y Planificación Física de la Administración Provincial de Artemisa?

En función del problema de investigación identificado y con el objetivo de alcanzar una solución satisfactoria se determina como **objeto de estudio** de la presente investigación los procesos de gestión de la información en base de datos relacionales y como **campo de acción** la persistencia y recuperación de los datos relacionales.

Para dar solución a lo anteriormente expuesto se precisa como **objetivo general** del presente trabajo:

- Desarrollar e implementar la capa de acceso a datos y la base de datos para los Módulos del CITMA y Planificación Física de la Administración Provincial de Artemisa.

Para cumplir con el objetivo propuesto se definen los siguientes **objetivos específicos** de la investigación:

- Elaborar la Fundamentación Teórica de la Investigación.
- Diseñar la capa de acceso a datos y la base de datos para los módulos de las direcciones del CITMA y Planificación Física.
- Implementar la capa de acceso a datos para los módulos de las direcciones del CITMA y Planificación Física.
- Validar los resultados obtenidos.

La **idea a defender** en el presente trabajo es que con el desarrollo de la capa de acceso a datos y la base de datos para la Dirección de CITMA y Planificación Física de la Administración Provincial de Artemisa, se contribuirá al correcto almacenamiento y recuperación de la información en estas direcciones.

Para lograr el cumplimiento de los objetivos señalados se definen las siguientes **tareas de la investigación:**

- Fundamentación del marco teórico de la investigación.
- Diseño de la capa de acceso a datos y la base de datos para los módulos de las direcciones del CITMA y Planificación Física.
- Implementación de la capa de acceso a datos para los módulos de las direcciones del CITMA y Planificación Física.
- Realización de pruebas funcionales a la capa de acceso del sistema de gestión.

Con el propósito de desarrollar las tareas planteadas, se utilizaron los métodos de investigación siguientes:

Métodos teóricos:

Histórico-Lógico: Este método es utilizado para estudiar cómo ha evolucionado y se han desarrollado las bases de datos desde su surgimiento hasta la actualidad, sus herramientas, formas de trabajos, modelos, entre otros. Realizando así el estudio de los procesos de gestión de la información de la Administración Provincial (AP).

Análisis y síntesis: Este método es utilizado para el análisis de la bibliografía referente al trabajo que se realizará. La confección del diseño teórico y metodológico de la investigación, análisis de las tendencias y tecnologías actuales de las bases de datos a nivel mundial.

Modelación: Este método es empleado para modelar las bases de datos para una mayor comprensión del trabajo que se realiza y los objetivos que se deben cumplir, dando respuesta a los requerimientos planteados con anterioridad.

Métodos empíricos:

Análisis documental: Se utilizó para consultar trabajos y buscar información relacionada con las bases de datos y la capa de acceso a datos lo que permitió elaborar la el marco teórico de la investigación.

Al finalizar el presente trabajo los involucrados en este proceso esperan obtener como **aporte práctico:**

- Capa de acceso a datos que garantice el correcto almacenamiento y la recuperación de la información de las Direcciones CITMA y Planificación Física de la Administración Provincial.

El presente trabajo está estructurado de la siguiente forma:

Capítulo 1. Fundamentación teórica: En este capítulo se ofrece una panorámica de las tendencias de las tecnologías relacionadas con las Bases Datos, así como su estado del arte a partir de modelos orientados a objetos; las tecnologías y metodología respectivamente adoptadas en el trabajo.

Capítulo 2. Diseño y arquitectura de la capa de acceso a datos: En este capítulo se definen los requisitos funcionales y no funcionales del sistema previamente definidos por el cliente, se hace un estudio de los patrones de diseño de bases de datos y se define la nomenclatura. Se representa el diagrama de clases persistentes, se detalla cada una de las clases y se diseña la BD.

Capítulo 3. Implementación y validación de la capa de acceso a datos: En este capítulo se realiza la implementación de la base de datos además de la validación. También se realizará un estudio para el análisis y eliminación de la redundancia de la información y seguridad de la BD.

Capítulo 1: Fundamentación teórica.

Introducción

En este capítulo se exponen los principales fundamentos teóricos en los que se basa la presente investigación. Se hace un estudio sobre los términos de capa de acceso a datos, bases de datos, sistemas gestores de base de datos así como sus principales características, ventajas y desventajas. Se aborda también sobre las metodologías y herramientas a utilizar para el desarrollo de la presente investigación.

1.1 Gestión de información de una base de datos.

Uno de los aspectos de mayor importancia en una base de datos (BD) es la gestión de información. Este aspecto abarca diferentes actividades como el almacenamiento, persistencia, recuperación, actualización y la eliminación de los datos.

La persistencia de datos hace referencia al tiempo en el que los datos existen y son utilizables (García-Bustelo, 2007). Por tanto se puede determinar que el término persistencia se refiere a mantener los valores de los datos durante todo su tiempo de vida. La recuperación por su parte permite acceder a los datos almacenados.

Las acciones mencionadas anteriormente se pueden realizar mediante las facilidades que brindan los Sistemas Gestores de Base de Datos (SGBD). Estos sistemas poseen lenguajes especiales llamados sublenguajes de datos (DSL, Data Sublanguage) que permiten manipular la información.

Los sublenguajes de datos se emplean para tratar los objetos de la BD y sus operaciones. Están compuestos por al menos dos lenguajes subordinados: “un lenguaje de definición de datos (DDL, Data Definition Language), el cual garantiza la definición o descripción de los objetos de la BD, y un lenguaje de manipulación de datos (DML, Data Manipulation Language), que garantiza la manipulación o tratamiento de esos objetos” (Mato, 1999). Algunos gestores incluyen un lenguaje

de control de los datos (DCL, Data Control Language) que permite controlar los permisos de los objetos de la BD.

Los sublenguajes de datos son empleados por las aplicaciones informáticas para manejar la información almacenada, llevando a cabo tareas específicas. Con este fin se implementan un conjunto de clases y funciones que conforman lo que se conoce como capa de acceso a los datos. Esta capa es la encargada de proveer a los programas de aplicación el acceso a los datos almacenados de forma persistente.

El empleo de distintas tecnologías de BD y las posibles sustituciones de una tecnología por otra, hacen necesario que las soluciones informáticas brinden soporte para distintos tipos de BD. Por estas razones se recomienda la implementación de una capa de abstracción que permita manipular la información con independencia del tipo de BD utilizada. La capa de abstracción de BD es una Interfaz de Programación de Aplicaciones (API, Applications Programming Interface) que permite la comunicación entre una aplicación informática y diferentes SGBD, abstrayendo las diferencias existentes entre estos sistemas (A. Otto, 2004)

Su utilización ofrece las siguientes ventajas:

- Reduce la complejidad: Libera a los desarrolladores de tratar con el mecanismo de persistencia de datos. Los desarrolladores ya no tendrán que preocuparse acerca de la procedencia de los datos, ni de la forma en que estos son almacenados; excepto cuando se desee lograr un alto rendimiento. (Nguèn, 2008)
- Portabilidad: Garantiza la independencia de las aplicaciones respecto a un tipo específico de tecnología de BD, facilitando la sustitución de una tecnología por otra sin tener que implementar o sobrescribir las consultas. (Army_Net_Centric_Data, 2009)
- Homogeneidad: Permite a los programadores utilizar una misma sintaxis para el acceso a los datos, independientemente del SGBD empleado. (Army_Net_Centric_Data, 2009)

- Reutilización: Constituye un componente reutilizable por distintas aplicaciones informáticas. Su utilización ahorra tiempo y esfuerzo durante el desarrollo de soluciones informáticas.

Una capa de abstracción de BD tiene las siguientes desventajas:

- Velocidad: Reduce la velocidad en dependencia de la cantidad de código adicional que tiene que ser ejecutado. Mientras más se abstraiga de las interfaces nativas de los SGBD, más lento será el rendimiento general. Por esta razón se debe analizar su utilización o no en entornos en los cuales se requiera un alto rendimiento.
- Dependencia: Proporciona una nueva dependencia funcional para un sistema informático ya que con el tiempo puede quedar obsoleta o no compatible.
- Operaciones enmascaradas: Puede limitar el número de operaciones disponibles de una BD a un subconjunto de las operaciones admitidas por las BD soportadas.

1.2 Importancia de la Capa de Acceso a Datos.

Actualmente, los equipos de desarrollo, en su mayoría, escogen arquitecturas multicapa para el desarrollo de aplicaciones Web, de modo que utilicen una CAD que encapsule todas las interacciones con la base de datos. Así, se simplifican las llamadas desde la capa de presentación, y permite encapsular la lógica de acceso a datos, siendo transparentes las operaciones de la CAD a la capa lógica del negocio. En la CAD es donde se implementan los componentes que interactúan con la BD y donde se encapsula el acceso a datos con estos componentes de una manera fácil, para lograr un rendimiento óptimo. Permite que las demás capas, componentes e interfaces no interactúen directamente con el servidor de BD y así excluir las complejidades del acceso a datos en el código fuente de otras capas. La CAD juega un importante papel dentro de la arquitectura de las aplicaciones, ella provee muchas ventajas claves entre las que se tienen:

- **Separación entre la capa de negocio y los gestores de BD:** los componentes de la CAD implementan aquellas funciones especializadas en el acceso a los diferentes servidores de BD y brindar esos datos, en un formato adecuado a la capa de negocio de la aplicación. Este nivel de aislamiento protege a los componentes de las otras capas, específicamente de la capa de negocio, de los cambios potenciales que puedan ocurrir en el servidor de BD.
- **Mejoras en el mantenimiento de la aplicación:** encapsula toda la gestión de acceso a datos en una sola capa, que ofrece la ventaja de reutilizar componentes, reduciendo la cantidad de código fuente a desarrollar y mantener.

1.3 Conceptos y características de Base de Datos.

Existen varias definiciones de las bases de datos, entre ellas se pueden citar las siguientes:

Es un conjunto exhaustivo no redundante de datos estructurados organizados independientemente de su utilización y su implementación en máquinas accesibles en tiempo real y compatible con usuarios concurrentes con necesidad de información diferente y no predicable en tiempo. (Rodríguez, 2004)

Es una colección de archivos interrelacionados, son creados con un DBMS (DataBase Manage System). El contenido de una base de datos engloba a la información concerniente (almacenadas en archivos) de una organización, de tal manera que los datos estén disponibles para los usuarios, una finalidad de la base de datos es eliminar la redundancia o al menos minimizarla. (Campoy, 2009)

Puede así definirse como una base de datos a conjunto de datos organizados y relacionados entre sí, permitiéndoles a los usuarios mantenerlos organizados, facilitando así la persistencia de los mismos durante un largo período de tiempo.

Las características que ha de presentar una base de datos de forma general son las siguientes:

- Control centralizado de los datos.
- Integridad de los datos.
- Minimización de las repeticiones.
- Independencia de los datos y las aplicaciones.
- Acceso concurrente a los datos.
- Coste mínimo de almacenamiento y mantenimiento.
- Versatilidad para la representación de relaciones.
- Establecimiento de medidas de seguridad.
- Facilidad para el cambio, actualización y optimización.

1.4 Clasificación de las bases de datos.

Teniendo en cuenta la variabilidad de los almacenados de las bases de datos pueden clasificarse en:

Bases de datos estáticas: Estas bases de datos se caracterizan por ser de solo lectura. Son utilizadas fundamentalmente para almacenar datos históricos que luego pueden ser utilizados en el estudio del comportamiento de un conjunto de datos a través del tiempo, permiten hacer proyecciones y tomar decisiones. Son bases de datos en las que sólo se puede consultar la información, no se puede modificar.

Bases de datos dinámicas: En estas BD la información que se almacena suele ser modificada con el tiempo. Se permiten operaciones como la modificación, la adición y eliminación de los datos además de las operaciones fundamentales de consulta.

Según el tipo de proyecto que se quiera desarrollar es que se debe entonces elegir el tipo de base de datos que se debe implementar. Para el desarrollo de la solución se seleccionó la BD dinámica por las características que posee, además de ser la que mejor se ajusta a las necesidades del usuario.

1.5 Fases del diseño de una base de datos.

El diseño de una base de datos se divide en tres fases fundamentales, tratando de descomponer el problema a resolver y garantizando una correcta ejecución de las metas de diseño definidas.

Diseño conceptual: En esta fase se propone un modelo conceptual sin tener en cuenta las consideraciones físicas. Esta primera fase consta de dos aspectos fundamentales: análisis de requisitos, centrando el trabajo en la definición de lo que se va a representar. El otro aspecto es la conceptualización, se representa lo antes definido.

Diseño lógico: Constituye un refinamiento del diseño conceptual, eliminando las construcciones que no son representables en el modelo escogido (relacional, orientado a objetos, entre otros). Esta fase parte de la anterior y tiene como objetivo principal obtener un modelo lógico eficiente en cuanto a operaciones de consulta y actualización.

Diseño físico: Es una traducción del esquema lógico al SGBD escogido. Considera las estructuras de almacenamiento y los métodos de acceso a la base de datos en memoria secundaria. Su objetivo es conseguir una implementación eficiente para esto se deben tener en cuenta las características del Sistema Operativo, el SGBD, el rendimiento y los requisitos de procesos, las características del hardware.

1.6 Modelos de Bases de Datos.

Con el desarrollo de las Bases de Datos también han evolucionado los tipos de modelado de las bases de datos. Todos los modelos de bases de datos tienen características propias que lo identifican teniendo en cuenta el contexto en el que surgieron. Los modelos de datos permiten describir y representar la realidad. Algunos de los modelos con frecuencia utilizados en las BD son:

- Modelo Jerárquico: fueron concebidas en los años 1960. Estas como su nombre lo indica, almacenan su información en una estructura jerárquica. En

este modelo los datos se organizan en una forma similar a un árbol al revés en donde un nodo padre de información puede tener varios hijos. El nodo que no tiene padres es llamado raíz, y a los nodos que no tienen hijos se los conoce como hojas. (Jahzeel, 2007)

Las BD jerárquicas son especialmente útiles en el caso de aplicaciones que manejan un gran volumen de información y datos muy compartidos permitiendo crear estructuras estables y de gran rendimiento. (Correa, 2010)

Una de las principales limitaciones de este modelo es su incapacidad de representar eficientemente la redundancia de datos.

- Modelo de Red: este es un modelo ligeramente distinto del jerárquico; su diferencia fundamental es la modificación del concepto de nodo, se permite que un mismo nodo tenga varios padres (posibilidad no permitida en el modelo jerárquico). (Jahzeel, 2007)

  También ofrece una gran ventaja sobre el modelo jerárquico, ya que presenta una solución eficiente al problema de redundancia de datos; pero, aun así, administrar la información en una BD de red ha significado que sea un modelo utilizado en su mayoría por programadores más que por usuarios finales.

- Modelo Relacional: este es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia, a diferencia de otros modelos como el jerárquico y el de red. Es más fácil de entender y de utilizar para un usuario esporádico de la BD. La información puede ser recuperada o almacenada mediante consultas que ofrecen una amplia flexibilidad y poder para administrar la información. (Jahzeel, 2007)

  Las bases de datos relacionales son el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. Su idea

fundamental es el uso de relaciones. Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados duplas.

- Modelo Orientados a Objetos: estas son capaces de almacenar tanto procesos como datos. Por este motivo las bases orientadas a objeto deben poder almacenar tanto información no convencional como imágenes estáticas o en movimiento, colecciones de sonidos, entre otros. Este tipo de BD deriva directamente de la llamada programación orientada a objetos, típica por ejemplo del lenguaje C/C++. (Schmieg, 2008)
- Modelo Deductivo: es un sistema de BD pero con la diferencia de que permite hacer deducciones a través de inferencias. Se basa principalmente en reglas y hechos que son almacenados en la BD. Las mismas son llamadas también BD lógicas, a raíz de que se basa en lógica matemática. (Jahzeel, 2007)
- Modelo Bases de Datos Documentales: son las derivadas de la necesidad de disponer de toda la información en el puesto de trabajo y de minimizar los tiempos de acceso a aquellas informaciones que se utilizan con frecuencia, no están estructuradas convenientemente. Esto se debe a que la procedencia de la información es muy variada: informes, notas diversas, periódicos y revistas. (Schmieg, 2008)
- Modelo de Bases de Datos Distribuida: un sistema de BD distribuidas se compone de un conjunto de nodos conectados entre sí mediante algún tipo de red de comunicaciones, en el cual cada sitio es un sistema de BD en sí mismo. Todos los sitios han convenido en trabajar juntos (si es necesario) con el fin de que un usuario de algún otro sitio pueda obtener acceso a los datos de otro punto de la red, tal como si todos los datos se encontrarán almacenados en el sitio propio del usuario. (Rozic, 2004)

Después de un análisis de los distintos modelos de BD se escoge el modelo de BD Relacional, debido a las ventajas que ofrece sobre los demás modelos analizados

anteriormente. Una de las ventajas que ofrece este modelo es el rápido entendimiento por parte de los usuarios, además la información puede ser recuperada o almacenada mediante consultas que ofrecen una amplia flexibilidad y poder para administrar la información.

1.7 Sistemas de Gestión de Bases de Datos.

El software de BD ha experimentado un auge extraordinario, además de rapidez y efectividad en los procesos. Con la existencia de múltiples entornos de desarrollo y la notable demanda de soluciones informáticas, han surgido multitud de gestores de BD.

Siendo los Sistemas Gestores de Base de Datos (SGBD) un tipo de software muy específico, dedicado a servir de interfaz entre la BD, el usuario y las aplicaciones que la utilizan y que prestan servicios para el desarrollo y el manejo de BD.

Dentro de los principales SGBD propietarios más potentes se encuentra Oracle y Microsoft SQL Server, y dentro del software libre, se tiene a PostgreSQL una opción muy completa.

Objetivos de un SGBD: (Gil, y otros, 2005)

Definir la BD mediante el lenguaje de definición de datos, el cual permite especificar la estructura, tipo y las restricciones sobre los datos, almacenándolo todo en la BD.

- Separar la descripción y manipulación de los datos, permitiendo un mayor entendimiento de los objetos, además de flexibilidad de consulta y actualización de los datos.
- Permitir la inserción, eliminación, actualización, consulta de los datos mediante el Lenguaje de Manejo de Datos, lo que permite resolver el problema que presentan los sistemas de archivos, donde hay que trabajar con un conjunto fijo de consultas o la necesidad de tener muchos programas de aplicaciones.
- Proporcionar acceso controlado a la base de datos.

- ✓ Seguridad: los usuarios no autorizados no pueden acceder a la base de datos.
- ✓ Integridad: mantiene la integridad y consistencia de la base de datos.
- ✓ Control de Recurrencia: permite el acceso compartido a la base de datos.
- ✓ Control de Recuperación: restablece la base de datos después de producirse un fallo de software o hardware.
- ✓ Diccionario de datos o Catálogo: contiene la descripción de los datos de la base de datos y es accesible por el usuario.

- ➢ Gestionar la estructura física de los datos y su almacenamiento, proporcionando eficiencia en las operaciones de la base de datos y el acceso al medio de almacenamiento.
- ➢ Proporcionar un mecanismo de vistas, que permita a cada usuario tener su propia vista o visión de la base de datos. El lenguaje de definición permite definir las vistas como subconjuntos de la base de datos, permitiendo:
  - ✓ Proporcionar un nivel de seguridad excluyendo datos para que no sean vistos por determinados usuarios.
  - ✓ Permiten que los usuarios vean los datos en el formato deseado.
  - ✓ Una vista representa una imagen consistente y permanente de la base de datos, aún cuando a la base de datos se le hagan cambios en su estructura.
- ➢ Eliminar la redundancia de datos, establecer una mínima duplicidad en los datos y minimizar el espacio en disco utilizado.
- ➢ Proveer interfaces procedimentales y no procedimentales, permitiendo la manipulación por usuarios interactivos y programadores.
- ➢ Independizar la estructura de la organización lógica de los datos

(Independencia física).

- Independizar la descripción lógica de la Base de datos y las descripciones particulares de los diferentes puntos de vistas de los usuarios.
- Permitir una fácil administración de los datos.

Ventajas y Desventajas de los SGBD.

Ventajas:

- Eliminan las inconsistencias en los datos.
- Permiten compartir los mismos datos entre diferentes aplicaciones con distintas necesidades.
- Ahorran espacio de almacenamiento al no existir redundancia o ser ésta escasa.
- Mejoran la seguridad de los datos, pues normalmente incorporan mecanismos de seguridad en el propio SGBD.
- Permiten la creación de entornos de alta disponibilidad, pues con algunos SGBD es posible llegar a disponer de aplicaciones funcionando ininterrumpidamente.
- Mejora en los servicios de copias de seguridad y de recuperación ante fallos.

Desventajas:

- Requieren una gran cantidad de espacio en disco y de memoria para trabajar de forma eficiente.
- Vulnerable a los fallos. El hecho de que todo esté centralizado en el SGBD hace que el sistema sea más vulnerable ante los fallos que puedan producirse.

Características de algunos SGBD.

Oracle

Es un sistema de gestión de base de datos fabricado por Oracle Corporation. Se considera como uno de los sistemas de bases de datos más completos que existe. Está herramienta se destaca por la estabilidad, la seguridad, la escalabilidad. Una de sus principales ventajas es que es una herramienta multiplataforma.

Tiene como principal característica desfavorable el enorme precio bajo la cual se comercializa (varios miles de euros). Aunque domina bastante el mercado, sufre la competencia de otras herramientas como: Microsoft SQL Server de Microsoft, PostgreSQL y MySql.

MySql

Es uno de los SGBD de más uso debido a su rapidez y facilidad de uso. Es fácil de instalar y configurar. Ofrece numerosas ventajas entre las que destacan las siguientes:

- Diseñado en vistas a la velocidad.
- Consume muy pocos recursos de CPU y memoria.
- Muy buen rendimiento.
- Tamaño del registro sin límite.
- Buena integración con PHP.
- Utilidades de administración (phpMyAdmin).
- Buen control de acceso usuarios-tablas-permisos.

Mysql posee dos desventajas fundamentales: no soporta subconsultas ni transacciones.

Desventajas:

- No es viable para su uso con grandes BD, a las que se acceda continuamente, ya que no implementa una buena escalabilidad.

- Carece de soporte para transacciones, subconsultas y re vertimiento de cambios realizados.

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.

Incluye características como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional.

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 caiga, existirá un registro de las transacciones a partir del cual se podrá 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.

1.8 Metodologías de Desarrollo de Software.

A partir de la necesidad de utilizar una serie de procedimientos, técnicas, herramientas y soporte para el desarrollo de un determinado software es que se da origen a las metodologías de desarrollo de software con el propósito de guiar a los desarrolladores del mismo.

Actualmente existen una gran variedad de metodologías para la creación de software los cuales son clasificadas en dos grupos, primeramente las **Metodologías Pesadas** las cuales son orientadas al control de los procesos donde se establece una rigurosidad en las actividades a desarrollar así como las herramientas y notaciones que se emplearán, y segundo las **Metodologías Ligeras o Ágiles** las cuales están orientadas a la interacción con el cliente mostrándoles versiones parcialmente funcionales del software al mismo en períodos de corto tiempo en las que el mismo pueda evaluar y seguir los cambios en el producto según su desarrollo, siendo esto último de forma incremental.

Metodología RUP

RUP (Rational Unified Process) es una de las metodologías pesadas más conocidas y utilizadas la cual divide el desarrollo en cuatro fases en los que se definen su ciclo de vida.

- **Trabajadores** (“quién”): Define el rol de un individuo, grupo de individuos, sistema automatizado o máquina, que trabajan en conjunto como un equipo. Ellos realizan las actividades y son propietarios de elementos.
- **Actividades** (“cómo”): Es una tarea que tiene un propósito claro, es realizada por un trabajador y manipula elementos.
- **Artefactos** (”qué”): Productos tangibles del proyecto que son producidos, modificados y usados por las actividades. Pueden ser modelos, elementos

dentro del modelo, código fuente y ejecutables.

- **Flujo de actividades** ("Cuándo"): Secuencia de actividades realizadas por trabajadores y que produce un resultado de valor observable.

En RUP se han agrupado las actividades en grupos lógicos definiéndose nueve flujos de trabajo, los seis primeros son flujos de ingeniería y los tres últimos de apoyo. Cada flujo de trabajo cumple con algunas actividades específicas. En el funcionan trabajadores específicos y producen y consumen artefactos también definidos. Cada fase representa un estado del proyecto, y produce un hito que sirve de entrada a la próxima fase. Todos los flujos se aplican en todas las fases, si bien algunos tienen más carga de trabajo que otros en algunas fases específicas. (Gallego, 2007)

Metodología SXP.

Esta metodología se encuentra en el grupo de las metodologías ágiles, surge en la Universidad de las Ciencias Informáticas y no es más que la unión de Scrum y XP.

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 el éxito del proyecto. (Peñalver, 2008)

Consta de cuatro 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 este listo para ser entregado.
- Entrega, puesta en marcha.
- Mantenimiento, donde se realiza el soporte para el cliente.

A partir de las facilidades que brinda la metodología SXP, se empleará para la realización de la presente investigación.

1.9 Herramientas Utilizadas.

Entre las herramientas empleadas para el desarrollo y administración de la base de datos de la presente investigación se encuentran las siguientes herramientas: NetBeans y Visual Paradigm.

NetBeans 7.0.1

Es una aplicación de código abierto (open source) diseñada para el desarrollo de aplicaciones, fácilmente portables entre las distintas plataformas (uso de la tecnología Java y un entorno de desarrollo integrado). Dispone de soporte para crear interfaces gráficas de forma visual, desarrollo de aplicaciones web, control de versiones, entre otras.

Es un programa con licencia CDDL4 que permite su uso libremente, brinda el código del mismo y es gratuita. Permite escribir, compilar, hacer un debug, ensamblar y desplegar aplicaciones, y aunque está escrito en Java, brinda soporte para toda clase de lenguajes de programación. Además, funciona en sistemas operativos compatibles con la máquina virtual Java (Windows XP, Vista, Windows 7, Ubuntu 9.10, Solaris, Mac OS X 10.5 o superior, entre otros). (Netbeans.org., 2012)

Visual Paradigm 6.4

Es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software, análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de mejor calidad y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Es multiplataforma y muy útil para la generación de código fuente en PHP. Incorpora el soporte para trabajo en equipo, proporcionando que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros de equipo. Esta

herramienta soporta BD que incluyen PostgreSQL, MySQL, Microsoft SQL Server, Oracle. (Infante, 2009)

1.10 Frameworks de desarrollo a utilizar.

Hibernate es un framework ORM (Modelo de Objeto Relacional) para la plataforma Java; distribuido bajo los términos de la licencia GNU LGPL ha ganado popularidad como herramienta de soporte a la capa de acceso a datos en el desarrollo de aplicaciones empresariales. Provee de un mapeo objeto-relacional básico, y otras características sofisticadas como caché y caché distribuida, carga perezosa y ávida.

Como todas las herramientas ORM, esta busca solucionar el problema de la diferencia entre dos modelos ampliamente utilizados para organizar y manipular datos: el orientado a objetos en las aplicaciones y el relacional en las bases de datos.

Para lograr esto el desarrollador debe especificar a Hibernate cómo es su modelo de datos. Con esta información Hibernate permite manipular los datos desde las aplicaciones operando sobre objetos con todas las características de la POO. Hibernate convierte los datos que define Java a los que define SQL, siendo transparente esta conversión para el desarrollador. Genera además las sentencias SQL y libera al desarrollador del manejo manual de los datos que resultan de la ejecución de dichas sentencias. También ofrece un lenguaje de consulta de datos llamado Hibernate Query Language (HQL).

Hibernate elimina la necesidad de parte del código de acceso a los datos, permitiendo que el desarrollador se concentre más en la lógica de negocio. Esta reducción de código representa un aumento de la productividad y permite que la capa de acceso a datos sea más entendible y fácil de mantener. (Hibernate.org., 2012)

Spring es un framework que provee amplio soporte para Hibernate. En particular, brinda implementaciones de DAO que ofrecen diversas utilidades para acceder a la session de Hibernate.

Algunas de las ventajas que brinda este framework al combinarse con algunas herramientas ORM son:

- **Manejo de sesión:** Hace de una forma más eficiente, sencilla y segura la forma en que se manejan las sesiones de varias herramientas ORM que se quiera utilizar.
- **Manejo de recursos:** Se puede manejar la localización y configuración de los SessionFactories de Hibernate o las fuentes de datos de JDBC, haciendo que estos valores sean más fáciles de modificar.
- **Facilidad de prueba:** Spring trata crear pequeños pedazos que se puedan aislar y probar por separado, ya sean sesiones o una fuente de datos (datasource).

1.11 Estado del Arte.

Las bases de datos son herramientas muy usadas en el mundo por las empresas debido a los grandes volúmenes de información que se manejan hoy en día. Además de que son herramientas muy útiles que permiten entre otras cosas favorecer la toma de decisiones en una institución.

En la actualidad son muchas las instituciones que cuentan con sistemas de bases de datos. Las instituciones, organizaciones o empresas que más aplican las bases de datos son las relacionadas con los bancos, las líneas aéreas, las telecomunicaciones, las finanzas, las ventas, la producción y los recursos humanos.

Bases de Datos en el Mundo.

GreenFILE: es una base de datos que ofrece información sobre el impacto de las acciones de los hombres en el medio ambiente. Esta aplicación ofrece resúmenes y

referencias bibliográficas de artículos, publicaciones académicas relacionadas con el tema.

ECOLEX: es una base de datos que tiene como objetivo fortalecer la preparación de las personas sobre el tema ambiental a nivel mundial, para esto pone a disposición una fuente de información bastante completa en materia de legislación ambiental.

Sistema de Información y Enlace Documental del Sistema Nacional Ambiental SIED-SINA (Colombia): Recoge y centraliza toda la información documental, ambiental, contenida en las bases de datos bibliográficas de las distintas entidades y organizaciones que conforman el Sistema Nacional Ambiental-SINA.

Centro mundial de datos sobre el Clima: Tiene la mayor base de datos sobre climatología. Esta base de datos gestiona 220 Tbytes de datos principalmente por el gran número de simulaciones del clima mundial que intenta predecir sus devastadores efectos en distintas zonas del planeta.

NERSC: El Centro de Computación Científica del Laboratorio Nacional de Energía. Cuenta con una base de datos que gestiona la información de una red de supercomputadoras, dedicada a investigación atómica, simulaciones astronómicas y experimentos de física de todo tipo.

En relación a la aplicación de bases de datos en los gobiernos se conoce que el Gobierno de los Estados Unidos cuenta con varias bases de datos para su trabajo entre ellas están Fed World que ofrece el servicio de localización en línea de la información diseminada por el gobierno federal de Estados Unidos, University of Michigan Document Center la cual contiene múltiples materiales gubernamentales a nivel local, estatal y federal, así como de gobiernos extranjeros y de organismos internacionales.

Bases de Datos a nivel Nacional.

Cuba, a pesar de sus limitaciones económicas se encuentra inmerso en la informatización del país. Muchas empresas cubanas ya han automatizado sus

procesos con el objetivo de lograr un trabajo de calidad con un ahorro de tiempo y esfuerzo.

Ejemplo de Bases de Datos Nacionales.

Sistema de Información Nacional para la Gestión de las Áreas Protegidas (SIGAP): Gestiona el manejo, control y seguimiento de la información relacionada con la infraestructura y biodiversidad existente en las áreas protegidas de Cuba. Este sistema cuenta con una base de datos confeccionada en MySQL.

Además existen otros casos como el Instituto Nacional de Recursos Hidráulicos (INRH) que cuenta con una base de datos elaborada en Microsoft Access para recopilar los datos hidrológicos y de la calidad del agua. También se puede observar casos como el Acuario Nacional de Cuba (ANC), el Centro de Inspección y Control Ambiental (CICA) y el Jardín Botánico Nacional entre muchos otros que usan este gestor de bases de datos.

Actualmente en la Facultad Regional Mártires de Artemisa se llevan a cabo proyectos vinculados a las Bases de Datos como el proyecto Sistema Informativo del CAP de Artemisa y Sistema Informativo del MINCEX (Ministerio del Comercio Exterior y la Inversión Extranjera).

Conclusiones Parciales.

En este capítulo se hace un análisis de los principales aspectos y conceptos relacionados con la investigación. Se realiza además un estudio de las posibles herramientas a utilizar. Como resultado de este análisis se determina lo siguiente:

- Es necesario el diseño de una base de datos y la implementación de una capa de acceso a datos, para contribuir al correcto almacenamiento y persistencia de los datos en las direcciones del CITMA y Planificación Física.
- Se seleccionó SXP como metodología de desarrollo.

- Se empleará como herramienta de modelado el Visual Paradigm en su versión 6.4 y como Entorno de Desarrollo Integrado Netbeans en su versión 7.0.1.
- Se hará uso de los Frameworks Hibernate y Spring para el desarrollo de la capa de acceso a datos de las direcciones CITMA y Planificación Física.

Capítulo 2: Diseño y arquitectura de la capa de acceso a datos.

Introducción

En el presente capítulo se definen los aspectos más importantes relacionados con el diseño de la capa de acceso a datos y la base de datos. Se realiza además un estudio del diagrama de clases persistente obtenido a partir del diagrama de clases del diseño por su importancia para la confección del modelo de datos. Se confecciona el diagrama entidad relación de la base de datos y se describen las tablas que la componen.

2.1 Diseño de la BD.

Las Bases de Datos necesitan de una definición de su estructura que le permita almacenar datos, reconocer el contenido, y recuperar la información. La puesta en práctica de la Base de Datos es el paso final en el desarrollo de aplicaciones de soporte del negocio. Para el diseño de la base de datos se realizó el diagrama entidad relación en el Visual Paradigm 6.4. En el diagrama entidad relación se especificaron los detalles físicos de la Base de Datos.

Normalización de los Datos. (Mato, 1999)**.**

La normalización se ha desarrollado para obtener estructuras de datos eficientes que eviten las anomalías de actualización, o sea, se basa en la necesidad de encontrar una representación del conjunto de relaciones para que el proceso de actualización sea más adecuado. Es la expresión formal del modo de realizar un buen diseño. Provee los medios necesarios para describir la estructura lógica de los datos en un sistema de información.

Existen diferentes niveles o fases de normalización, las cuales son: Primera Forma Normal (1FN), Segunda Forma Normal (2FN), Tercera Forma Normal (3FN), Forma Normal de Boyce-Codd (FNBC) y existen además 4FN y 5FN.Cada fase supone que se ha concluido la anterior.

Primera Forma Normal (1FN)

1. Una relación está en (1FN) si cumple la propiedad de que sus dominios no tienen elementos que, a su vez, sean conjuntos.

2. Toda relación normalizada, o sea, con valores atómicos de los atributos.

3. La relación no incluye ningún grupo repetitivo.

Segunda Forma Normal (2FN)

Una relación se dice que está en 2FN si está en 1FN y si, y sólo si, los atributos no llaves (ni primarias, ni candidatas), son funcional y completamente dependientes de la llave primaria de dicha relación.

Tercera Forma Normal (3FN)

Una relación está en 3FN si está en 2FN y si, y sólo si, los atributos no llaves son independientes de otro atributo no llave primaria. Esto es lo mismo que decir que se deben eliminar las dependencias transitivas de atributos no llaves respecto a la llave primaria, estando ya la relación en 2FN.

Diagrama Entidad – Relación.

El modelo conceptual más utilizado para el diseño de BD es el diagrama entidad-relación, capaz de mostrar una visión más amplia de como trabajará el sistema y la información a almacenar en el mismo, además expresa entidades relevantes para un sistema de información, sus interrelaciones y propiedades. Los diagramas entidad-relación son una necesidad para diseñar y mantener una buena BD relacional. Son una forma gráfica para representar una Base de Datos.

En la dirección del CITMA se representaron un total de 48 tablas, de ellas 9 son nomencladores y en la dirección de Planificación Física se representaron un total de 8 tablas de ellas solamente se identificó un nomenclador.

Para el diseño del diagrama entidad-relación, fue necesario establecer un estándar para organizar el trabajo. Para esto, se determinó nombrar las tablas con la letra

inicial (d) seguido del (nombre del modelo) y por último el (nombre de la dirección) como se puede apreciar en el siguiente ejemplo:

dfctcitma (d (representa una tabla) fct (representa el modulo del fórum de ciencia y técnica) citma (representa la dirección)).

ddenominacionpfisica (d (representa una tabla) denominacion (representa el nombre del modelo) pfisica (representa la dirección)).

Los nomencladores se representaron con la letra inicial (n) seguido del (nombre del nomenclador) y por último del (nombre de la dirección) como se muestra en el siguiente ejemplo:

ncriteriocitma (n (representa un nomenclador) criterio (representa los criterios) citma (representa la dirección)).

Descripción de las tablas del DER.

A continuación se describen algunas de las tablas del DER de la dirección del CITMA (Anexo 1), para una mayor comprensión de la misma ir al artefacto Modelo Canónico de los Datos:

Nombre:	dmodulocitma		
Descripción:	Tabla principal que almacena los datos en común de los módulos.		
Campo	**Tipo de Dato**	**Tamaño**	**Descripción**
nold	Serial	999	Identificador del módulo.
id_municipio	Serial	999	Identificador del municipio.
fecha	Cadena de texto	10	Fecha del módulo.
di_dependencia	Cadena de texto	20	Dependencia que emite el módulo.
di_direccion	Cadena de texto	30	Dirección.
di_telefono	Entero	6	Teléfono.
di_email	Cadena de texto	30	Correo electrónico.
di_periodinforma	Cadena de texto	20	Período que informa.

di_nombinforma	Cadena de texto	20	Nombre del informante.

Tabla 1. Descripción de la tabla módulo.

Nombre:	dempinscriptascitma		
Descripción:	Tabla de las empresas inscriptas.		
Campo	**Tipo de Dato**	**Tamaño**	**Descripción**
idempinsc	Serial	999	Identificador de las empresas.
idmmedamb	Entero	10	Identificador del módulo de medio ambiente.
empresas	Cadena de Texto	20	Nombre de la empresa.
organismo	Cadena de Texto	10	Nombre del organismo.

Tabla 2. Descripción de la tabla de las Empresas Inscriptas.

Nombre:	dareasprotegidascitma		
Descripción:	Tabla de las áreas protegidas.		
Campo	**Tipo de Dato**	**Tamaño**	**Descripción**
idareaprot	Serial	999	Identificador del área.
idmmedamb	Entero	10	Identificador del módulo de medio ambiente.
idcateg	Entero	10	Identificador de la categoría.
nomb_areaprote g	Cadena de texto	50	Nombre del área.
extens_km	Double	10	Extensión del área.
administra	Cadena de texto	20	Nombre del quien la administra.
conplandmanejo	Boolean	1	Plan de manejo.

Tabla 3. Descripción de la tabla de las Áreas Protegidas.

A continuación se describen algunas de las tablas del DER de la dirección de Planificación Física (Anexo 2), para una mayor comprensión de la misma ir al artefacto Modelo Canónico de los Datos:

Nombre:	dorganismoinfractpfisica		
Descripción:	Tabla de los organismos infractores.		
Campo	**Tipo de Dato**	**Tamaño**	**Descripción**
idorg	Serial	999	Identificador del organismo.
cod_org	Entero	10	Código del organismo.
org_infract	Cadena de texto	30	Nombre del organismo infractor.
descrp_ileg	Cadena de texto	50	Descripción de la ilegalidad.
nomb_invers	Cadena de texto	50	Nombre de la inversión.
oblig_hacer	Cadena de texto	50	Obligación hacer.
observ	Cadena de texto	100	Observaciones.
fecha	Cadena de texto	10	Fecha.
fechact	TimesTamp	10	Fecha de actualización.

Tabla 4. Descripción de la tabla de los Organismos Infractores.

Nombre:	ddenominacionpfisica		
Descripción:	Tabla de las denominaciones.		
Campo	**Tipo de Dato**	**Tamaño**	**Descripción**
cod_denom	Entero	10	Código de la denominación.
nomb_denomin	Cadena de texto	30	Nombre de la denominación.
permis_micro	Boolean	1	Permisología tipo micro.
permis_lic	Boolean	1	Permisología tipo licencia.
fecha	Cadena de texto	10	Fecha.
fechact	TimesTamp	10	Fecha de actualización.

Tabla 5. Descripción de la tabla de Denominación.

Nombre:	davanceejecinverspfisica		
Descripción:	Tabla de avance de ejecución de inversiones.		
Campo	**Tipo de Dato**	**Tamaño**	**Descripción**
idavance	Serial	999	Identificador del avance
id_municipio	Entero	10	Identificador del municipio.
cod_denom	Entero	10	Código de la denominación.
porc_mov_tierra	Double	10	Porciento del movimiento de tierra.
porc_ciment	Double	10	Porciento de cimentación.
porc_estruc	Double	10	Porciento de estructura.
porc_termin	Double	10	Porciento de terminación.
cant_cert_utiliz	Entero	10	Cantidad de certificados utilizados.
fecha	Cadena de texto	10	Fecha.
fechact	TimesTamp	10	Fecha de actualización.

Tabla 6. Descripción de la tabla de Avance de Ejecución de Inversiones.

2.2 Clases Persistentes.

Las clases persistentes son las clases que necesitan ser capaz de guardar su estado en un medio permanente. Son aquellas que contienen toda la información valiosa e importante que necesariamente debe ser almacenada en la BD.

El diagrama de clases persistentes se utiliza para modelar la estructura lógica de la BD, donde las tablas son representadas por clases y las columnas por atributos de clases. Las clases persistentes y sus atributos hacen referencia directamente a las entidades lógicas y a sus atributos.

El diagrama de ambas direcciones fue realizado de manera automática con la herramienta Visual Paradigm a partir de la conexión establecida con la base de datos.

Descripción de las clases persistentes.

A continuación se describen algunas de las clases persistentes pertenecientes a la dirección del CITMA, para una mayor comprensión de la misma ir al artefacto Modelo Canónico de los Datos:

Clase:	Dmodulocitma			
Descripción:	Clase modulo.			
Tipo:	Superclase			
Roles o Propiedades	**Descripción**	**Valor Requerido**	**Llave**	**Relación**
noId	Identificador del módulo.	Serial	X	dfctcitma, dplanificacioncitma ,dactivregulatcitma ,dtecnologiacitma, dcienciacitma, dmedioambcitma
id_municipio	Identificador del municipio.	Serial	X	Nmunicipios
fecha	Fecha del módulo.	Cadena de texto		
di_dependencia	Dependencia que emite el módulo.	Cadena de texto		
di_direccion	Dirección.	Cadena de texto		
di_telefono	Teléfono.	Entero		
di_email	Correo electrónico.	Cadena de texto		
di_periodinforma	Período que informa.	Cadena de texto		
di_nombinforma	Nombre del informante.	Cadena de texto		

Tabla 7. Descripción de la clase Dmodulocitma.

Clase:	Dempinscriptascitma			
Descripción:	Clase empresas inscriptas.			
Tipo:	Superclase			
Roles o Propiedades	**Descripción**	**Valor Requerido**	**Llave**	**Relación**
idempinsc	Identificador de las empresas.	Serial	X	
idmmedamb	Identificador del módulo de medio ambiente.	Entero	X	dmmedambcitma
empresas	Nombre de la empresa.	Cadena de Texto		
organismo	Nombre del organismo.	Cadena de Texto		

Tabla 8. Descripción de la clase Dempinscriptascitma.

Clase:	Dareasprotegidascitma			
Descripción:	Clase áreas protegidas.			
Tipo:	Superclase			
Roles o Propiedades	**Descripción**	**Valor Requerido**	**Llave**	**Relación**
idareaprot	Identificador del área.	Serial	X	
idmmedamb	Identificador del módulo de medio ambiente.	Entero	X	dmmedambcitma
idcateg	Identificador de la categoría.	Entero	X	ncategoriacitma
nomb_areaprotog	Nombre del área.	Cadena de texto		
extens_km	Extensión del área.	Double		
administra	Nombre del quien la administra.	Cadena de texto		
conplandmanejo	Plan de manejo.	Boolean		

Tabla 9. Descripción de la clase Dareasprotegidascitma.

A continuación se describen algunas de las clases persistentes pertenecientes a la dirección de Planificación Física, para una mayor comprensión de la misma ir al artefacto Modelo Canónico de los Datos:

Clase:	Dorganismoinfractpfisica			
Descripción:	Clase organismos infractores.			
Tipo:	Superclase			
Roles o Propiedades	**Descripción**	**Valor Requerido**	**Llave**	**Relación**
idorg	Identificador del organismo.	Serial	X	dresumenilegalidpfisica
cod_org	Código del organismo.	Entero		
org_infract	Nombre del organismo infractor.	Cadena de texto		
descrp_ileg	Descripción de la ilegalidad.	Cadena de texto		
nomb_invers	Nombre de la inversión.	Cadena de texto		
oblig_hacer	Obligación hacer.	Cadena de texto		
observ	Observaciones.	Cadena de texto		
fecha	Fecha.	Cadena de texto		
fechact	Fecha de actualización.	TimesTamp		

Tabla 10. Descripción de la clase Dorganismoinfractpfisica.

Clase:	Ddenominacionpfisica			
Descripción:	Clase denominaciones.			
Tipo:	Superclase			
Roles o Propiedades	**Descripción**	**Valor Requerido**	**Llave**	**Relación**
cod_denom	Código de la denominación.	Entero	X	
nomb_denomin	Nombre de la denominación.	Cadena de texto		
permis_micro	Permisología tipo micro.	Boolean		
permis_lic	Permisología tipo licencia.	Boolean		
fecha	Fecha.	Cadena de texto		

fechact	Fecha de actualización.	TimesTamp		

Tabla 11. Descripción de la clase Ddenominacionpfisica.

Clase:	Davanceejecinverspfisica			
Descripción:	Clase avance de ejecución de inversiones.			
Tipo:	Subclase			
Roles o Propiedades	**Descripción**	**Valor Requerido**	**Llave**	**Relación**
idavance	Identificador del avance	Serial	X	
id_municipio	Identificador del municipio.	Entero	X	nmunicipios
cod_denom	Código de la denominación.	Entero	X	ddenominacion pfisica
porc_mov_tierra	Porciento del movimiento de tierra.	Double		
porc_ciment	Porciento de cimentación.	Double		
porc_estruc	Porciento de estructura.	Double		
porc_termin	Porciento de terminación.	Double		
cant_cert_utiliz	Cantidad de certificados utilizados.	Entero		
fecha	Fecha.	Cadena de texto		
fechact	Fecha de actualización.	TimesTamp		

Tabla 12. Descripción de la clase Davanceejecinverspfisica.

2.3 Descripción de la arquitectura.

Según la IEEE la Arquitectura de Software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución.

La presente solución está basada en la arquitectura en tres capas:

- La capa de presentación: En esta capa se diseña todo lo que constituye la interfaz gráfica y la interacción del usuario con el sistema.

- La capa de negocio: Contiene todas las subrutinas creadas con el propósito de regular alguna acción del usuario.
- La capa de acceso a datos: En esta capa se programa todo lo que tiene que ver con el acceso a la base de datos. Esta capa queda encargada de tomar la información de la base de datos dada una petición de la capa del negocio, que a su vez es generada por la capa de presentación.

Este trabajo se centra específicamente en la capa de acceso a datos. En esta capa se encuentra todo lo relacionado a la gestión y la persistencia de los datos así como todos los métodos que serán utilizados en la capa de negocio estableciéndose de esta forma la comunicación entre ambas capas.

2.4 Estructura y elementos de paquete.

La capa de persistencia o de acceso a datos encapsula el conjunto de clases y componentes responsables de almacenar y recuperar datos, incluyendo la integración directa de la aplicación con la base de datos; una capa de persistencia es la base de la arquitectura en capas.

Desde el entorno de desarrollo se muestra la siguiente estructura de paquete:

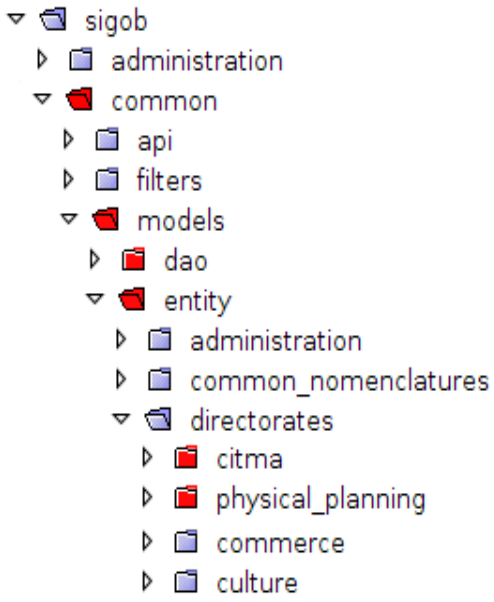


Imagen 1. Estructura de la aplicación.

La capa de acceso a datos de las direcciones del CITMA y Planificación Física se encuentra en el paquete models donde a su vez este contiene el paquete dao y entity, almacenando este último las diferentes direcciones que componen la Administración Provincial de Artemisa.

A continuación se describen los componentes que se encuentran en estas carpetas:

common.models.dao

Archivo	Descripción
GenericDao.java	Hereda los métodos de la clase HibernateDaoSupport.

Tabla 13. Descripción del componente dao.

common.models.entity.citma

Archivo	Descripción
Dactivnuclearcitma.java	Describe los atributos de la Actividad Nuclear.
Dactivregulatcitma.java	Describe los atributos de la Actividad Regulatoria.
Daplicdecretoleycitma.java	Describe los atributos de la Aplicación del Decreto Ley.
Dareasprotegidascitma.java	Describe los atributos de las Áreas Protegidas
Dcienciacitma.java	Describe los atributos del modulo de Ciencia.
Dejecacumcitma.java	Describe los atributos de la Ejecución Acumulada.
Dejecacumtmcitma.java	Describe los atributos de la Ejecución Acumulada Turquino Manatí.
Dejecfinancpctturqmanaticitma.java	Describe los atributos de la Ejecución Financiada Plan Ciencia y Técnica Turquino Manatí.
Dejecfinancplancienctecncitma.java	Describe los atributos de la Ejecución Financiada Plan Ciencia y Técnica.
Dempinscriptascitma.java	Describe los atributos de las Empresas Inscriptas.
Destejecprocitma.java	Describe los atributos del Estado de Ejecución de Proyectos.
Destejeproyaprobcontfinfondmedambcitma.java	Describe los atributos del Estado de Ejecución Proyecto Aprobado Contra Financiamiento Fondo Medio Ambiente.
Destotorglicambimposmultcitma.java	Describe los atributos del Estado de Otorgamiento Licencias Ambientales

	Multadas.
Dfctcitma.java	Describe los atributos del módulo del Fórum de Ciencia y Técnica.
Dinformgeneralizcitma.java	Describe los atributos del Informe Generalizado.
Dinfresulgralcitma.java	Describe los atributos del Informe de Resultado General.
Dinspccambcitma.java	Describe los atributos de las Inspecciones Ambientales.
Dinsplicambotorgcitma.java	Describe los atributos de las Inspecciones de Licencias Ambientales Otorgadas.
Dinspreinspbiologcitma.java	Describe los atributos de las Inspecciones y Re inspecciones Biológicas.
Dinvercitma.java	Describe los atributos de las Inversiones.
Dliberacreportmedambcitma.java	Describe los atributos de la Liberación de Reporte Medio Ambiente.
Dlistproyadministrocitma.java	Describe los atributos de la Lista de Proyectos que Administro.
Dlugvisitcitma.java	Describe los atributos de los Lugares Visitados.
Dmedioambcitma.java	Describe los atributos del módulo de Medio Ambiente.
Dmodulocitma.java	Describe los atributos del Módulo.
Dobrascientifcitma.java	Describe los atributos de las Obras Científicas.
Dplancitma.java	Describe los atributos del Plan.
Dplanificacioncitma.java	Describe los atributos del módulo de Planificación.
Dprogproytejeccitma.java	Describe los atributos de los Programas y Proyectos en Ejecución.
Dproaprobpendcontratfondmedambcitma.java	Describe los atributos de los Proyectos Aprobados Pendientes a Contrato Fondo Medio Ambiente.
Dproycitma.java	Describe los atributos de los Proyectos.
Dpturqmanatcitma.java	Describe los atributos del Plan Turquino Manatí.
Dreduccargascontamcitma.java	Describe los atributos de Reducción de Cargas Contaminantes.
Dreinspambcitma.java	Describe los atributos de las Re inspecciones Ambientales.
Dresulalcanzcitma.java	Describe los atributos de los Resultados Alcanzados.

Dsalidfinejecplancienctecncitma.java	Describe los atributos de las Salidas Finales Ejecución Plan Ciencia y Técnica.
Dsolicitlicambcitma.java	Describe los atributos de la Solicitud de Licencia Ambiental.
Dtecncitma.java	Describe los atributos del módulo de Tecnología.
Dtecnologiacitma.java	
Nactividadcitma.java	Nomenclador en el que se almacena las Actividades.
Ncaractercitma.java	Nomenclador en el que se almacena los Carácter.
Ncategoriacitma.java	Nomenclador en el que se almacena las Categorías.
Nclasifinspcitma.java	Nomenclador en el que se almacena las Clasificaciones.
Ncriteriocitma.java	Nomenclador en el que se almacena los Criterios.
Nestadoejecitma.java	Nomenclador en el que se almacena el Estado de Ejecución.
Noacecitma.java	Nomenclador en el que se almacenan los Organismos del Consejo de Estado.
Nresultadocitma.java	Nomenclador en el que se almacena los Resultados.
Ntrimestrecitma.java	Nomenclador en el que se almacena los Trimestres.

Tabla 14. Descripción del componente citma.

common.models.entity.phisical_planning

Archivo	Descripción
Davanceejecinverspfisica.java	Describe los atributos del Avance de Ejecución de Inversiones.
Dcertifmedlinderospfisica.java	Describe los atributos de la Certificación de Medidas y Linderos.
Ddenominacionpfisica.java	Describe los atributos de la Denominación.
Ddinamicasenthumpfisica.java	Describe los atributos de la Dinámica de Sentamiento Humano.
Dorganismoinfractpfisica.java	Describe los atributos de los Organismos Infractores.
Dresumenilegalidpfisica.java	Describe los atributos del Resumen de

Dusoocupacsulelodecrt259pfisica.java	llegalidad. Describe los atributos del Uso Ocupacional del Suelo Decreto Ley 259.

Tabla 15. Descripción del componente phisica_planing.

Conclusiones Parciales.

En este capítulo se especificaron los principales aspectos relacionados con el diseño y la arquitectura de la solución propuesta:

- Se describió como queda definida la arquitectura de la solución.
- Se crearon los diagramas de clases persistentes y el modelo entidad-relación.
- Se realizó la descripción de las principales entidades, logrando obtener finalmente la construcción de la base de datos propuesta.
- Se definieron los elementos y la estructuración que debe tener la capa de persistencia, indicando la distribución lógica de cada componente desde el entorno de desarrollo integrado.

Capítulo 3: Implementación y validación de la capa de acceso a datos.

Introducción.

Este capítulo contiene una descripción de los elementos referentes a la implementación de la capa de acceso a datos para los módulos de las direcciones de CITMA y Planificación Física. También incluye los aspectos relacionados con la validación de la capa de acceso a datos teniendo en cuenta aspectos de particular importancia, así como la descripción de los pasos y archivos generados para la generación de los POJOS.

3.1 Implementación DAO Genérico.

Para gestionar las entidades en ambas direcciones, con el fin de tener independencia de la capa de datos, se implementó el patrón DAO que encapsula el acceso a la base de datos. Por lo que cuando la capa de lógica de negocio necesite interactuar con esta, va a hacerlo a través de la API que le ofrece DAO. En la aplicación se implementó una clase genérica que hereda de la clase HibernateDaoSupport implementada en el framework spring.

HibernateDaoSupport ofrece la ventaja de trabajar con las secciones de Hibernate de forma automática y sin ningún tipo de gestión por parte del administrador de bases de datos. En la clase GenericDao fueron creadas las funciones necesarias para trabajar de forma general con todas las entidades.

Dentro de las funciones que presenta la clase GenericDao se encuentran:

save: Salva el objeto pasado por parámetro.

```
public void save(Object entity) {
    getHibernateTemplate().save(entity);

    //Other Way
    //getHibernateTemplate().persist(entity);
}
```

Imagen 2. Método save.

saveOrUpdate: Este método recibe como parámetro un objeto, verifica que el objeto este almacenado en la base de datos, si es encontrado lo actualiza en caso

contrario es salvado.

```
public void saveOrUpdate(Object entity) {
    getHibernateTemplate().saveOrUpdate(entity);

}
```

Imagen 3. Método saveOrUpdate.

find: Retorna una lista de todas las tuplas de la base de datos mapeadas como objetos, se le pasa como parámetro el tipo de dato que se desea buscar.

```
public <T> List<T> find(Class<T> entityClass) {
    return getHibernateTemplate().loadAll(entityClass);
}
```

Imagen 4. Método find.

findByPK: Busca y retorna un objeto a través de la llave primaria y el tipo.

```
public <T> Object findByPK(Class<T> entityName, Serializable id) {
    return (Object) getHibernateTemplate().get(entityName, id);
}
```

Imagen 5. Método findByPK.

findByParamAndValue: Devuelve la lista de objetos que cumplan con una condición a partir del nombre de un campo y su valor.

```
public <T> List<T> findByParamAndValue(Class<T> entityClass, String param,
        Object value) {

    DetachedCriteria criteria = DetachedCriteria.forClass(entityClass);
    criteria.add(Restrictions.eq(param, value));
    return getHibernateTemplate().findByCriteria(criteria);
}
```

Imagen 6. Método findByParamAndValue.

findByParamsAndValues: Retorna todos los objetos que cumplan con una serie de atributos y valores pasados por parámetro.

```
public <T> List<T> findByParamsAndValues(Class<T> entityClass,
        String[] params, Object[] values) {

    List<T> list = new ArrayList<T>();
    if (params.length > 0 && values.length > 0) {
        DetachedCriteria criteria = DetachedCriteria.forClass(entityClass);
        criteria.add(Restrictions.eq(params[0], values[0]));

        if (params.length > 1 && values.length > 1) {
            for (int i = 1; i < params.length; i++) {
                criteria.add(Restrictions.eq(params[i], values[i]));
            }
        }

        list = getHibernateTemplate().findByCriteria(criteria);
    }

    return list;
}
```

Imagen 7. Metodo findByParamsAndValues.

update: Actualiza una entidad determinada pasada por parámetro.

```
public void update(Object entity) {
    getHibernateTemplate().update(entity);
}
```

Imagen 8. Método update.

delete: Elimina un objeto pasado por parámetro.

```
public void delete(Object entity) {
    getHibernateTemplate().delete(entity);
}
```

Imagen 9. Método delete.

deleteAll: Elimina todas las tuplas de una tabla recibiendo como parámetro el tipo de esta.

```
public <T> void deleteAll(Collection<T> collection) {
    getHibernateTemplate().deleteAll(collection);
}
```

Imagen 10. Método deleteAll.

3.2 Mapeo Objeto Relacional (ORM).

La conexión entre la aplicación y la base de datos es solucionada mediante el mapeo objeto relacional que proporciona Hibernate. Este presenta un conjunto de herramientas que facilitan la realización de la ingeniería inversa y la generación de código.

ORM es una técnica que ofrece un mapeo automatizado y transparente, además de las muchas posibilidades de optimización que no serían posible en el desarrollo de una capa de persistencia, donde la totalidad de la codificación se generará manualmente. Este permite generar el archivo de configuración para Hibernate, en el cual se especifica dentro de los elementos <hibernate-configuration> y <session-factory>, el dialecto de la base de datos, el driver para la conexión (JDBC), la cadena de conexión que contiene el servidor y el nombre de la base de datos a la cual se hará la conexión, incluyendo la autenticación de acceso al servidor de base de datos.

Mapeo de las Entidades.

La generación de los archivos POJOS se realizó en el IDE NetBeans 7.0.1 en el cual se encuentran asistentes que permiten crear las clases POJO con solo configurar la conexión a la base de datos.

Para mapear las clases a partir de la base de datos es necesario realizar una secuencia de pasos, creando primeramente el archivo de configuración <hibernate.cgf.xml> donde en el mismo es especificado el IP de conexión de la BD y el nombre de la misma, así como el usuario y la contraseña.

```
<hibernate-configuration>
    <session-factory>
        <property name="hibernate.dialect">
            org.hibernate.dialect.PostgreSQLDialect
        </property>
        <property name="hibernate.connection.driver_class">
            org.postgresql.Driver
        </property>
        <property name="hibernate.connection.url">
            jdbc:postgresql://10.208.1.250:5432/sigob
        </property>
        <property name="hibernate.connection.username">user</property>
        <property name="hibernate.connection.password">user</property>
        <property name="hibernate.show_sql">true</property>
        <property name="hibernate.query.factory_class">
            org.hibernate.hql.classic.ClassicQueryTranslatorFactory
        </property>
```

Imagen 11. Archivo de configuración de Hibernate.

Luego se procede a generar el archivo de ingeniería inversa <hibernate.reveng.xml> donde se encuentran los nombres de las tablas de la BD.

```
<hibernate-reverse-engineering>
  <schema-selection match-catalog="sigob" match-schema="mod_citma"/>
  <table-filter match-name="dprogproytejeccitma"/>
  <table-filter match-name="dcumplnomenclnacimpact"/>
  <table-filter match-name="nresultadocitma"/>
  <table-filter match-name="dinspccambcitma"/>
  <table-filter match-name="dinfresulgralcitma"/>
```

Imagen 12. Archivo de ingeniería inversa de Hibernate.

Una vez configurado estos dos archivos se procede a crear los POJOS. Hibernate presenta dos formas de crearlos, con archivos XML y con anotaciones. Con archivos XML se tiene el archivo POJO junto a un archivo XML que sirve de intermediario para enlazar cada atributo de la clase con su equivalente en la base de datos. El mapeo a través de anotaciones se realiza dentro de la misma entidad, especificándole por medio de metadatos los atributos que tienen su equivalente en la base de datos.

Este permite reducir la cantidad de líneas de código, y además la cantidad de archivos necesarios para el mapeo, lo que simplifica tanto el esfuerzo en codificar, mantener, como la probabilidad de cometer errores. Es común que al realizar

proyectos complejos la cantidad de componentes que intervienen produzcan un exceso de archivos de configuración, los que dificultan el mantenimiento.

En la aplicación se decidido mapear por medio de anotaciones ya que es un método más sencillo y no genera tantos ficheros cuando se manejan grandes cantidades de entidades, quedando las entidades mapeadas como se muestra a continuación de forma general:

```
1.  @Entity
2.  @Table(name="Nombre de la tabla"
3.  ,schema="Nombre del esquema"
4.  )
5.  public class Nombre de la clase  implements java.io.Serializable {
6.
7.  /*Atributos de la clase*/
8.  private long  id;
9.
10. public Constructor() {
11. }
12. /*Métodos Get y Set de cada uno de los atributos junto los metadatos
    correspondiente*/
13. }
```

Imagen 13. Estructura general de los POJOS en Hibernate.

Describiendo a continuación algunas de las anotaciones utilizadas:

- **@Entity:** A partir de esta anotación se indica a Hibernate que la clase es persistente.
- **@Table**(name = "Nombre de la Tabla", schema="Nombre del esquema"): A través de esta anotación se especifica un nombre de tabla así como el del esquema, en caso de diferir al de la clase. De no indicarlo, Hibernate tomará como nombre de la tabla asociada, el mismo que el de la clase.
- **@Column:** que es la anotación estándar JPA para definir el nombre de la columna en la base de datos asociada a esta columna. Si no se define esta anotación, el nombre de la columna será igual al nombre de la propiedad. El nombre de la propiedad puede derivarse a partir de un método get o set.

- **@Id:** A través de esta anotación se indica que la property mapeada será la llave primaria de la tabla.
- **@GeneratedValue:** es la anotación estándar para definir estrategias para la generación de llaves primarias. En este caso se hace uso de un campo de identidad. Hibernate define facilidades adicionales al estándar para definir estrategias de generación de identidad; estas facilidades se pueden definir por medio de anotaciones propias de Hibernate.
- **@ManyToOne:** Indica la relación que existe entre las entidades mapeadas. Esta anotación puede estar acompañada de ciertos parámetros, como el tipo de fetching y el cascading**.**

3.3 Validación teórica del diseño.

Para efectuar un buen diseño de una base de datos se deben tener en cuenta varios aspectos significativos como: la integridad de los datos, la redundancia de la información.

Análisis de la integridad de los datos.

La integridad de los datos se refiere a la validez, corrección, consistencia y completitud de los datos almacenados en la BD. Tiene como función proteger la BD contra operaciones que introduzcan inconsistencias en los datos. Un control de integridad o restricciones es aquel que permite definir con precisión el rango de valores válidos para un elemento y/o las operaciones que serán consideraciones válidas en la relación de tales elementos.

Cuando se realizan acciones como insertar datos incorrectos o no válidos, modificar la información existente tomando un valor equivocado o eliminando datos que produzcan la violación de alguna restricción en la BD, se está ante una situación en la que la integridad de los datos se encuentra afectada.

Categorías de integridad de los datos:

Al definir cada atributo sobre un dominio se impone una restricción sobre el conjunto de valores permitidos para cada atributo. A este tipo de restricciones se les denomina restricciones de dominio. Hay además dos reglas de integridad muy importantes que son restricciones que se deben cumplir en todas las BD relacionales y en todos sus estados o instancias, las reglas se deben cumplir todo el tiempo. Estas reglas son la de integridad de entidades y la de integridad referencial. Antes de definirlas es preciso conocer los conceptos de nulo y dominio.

- **Nulo:** es un indicador que le dice al usuario que dato falta o no es aplicable. Por conveniencia, un dato que falta normalmente se dice que tiene valor nulo, pero el valor de nulo no es un valor de dato real. En vez de eso es una señal o un recordatorio de que el valor falta o es desconocido.
- **Dominio:** posibles valores que puede tener un campo. Un dominio no es más que un tipo de dato; posiblemente un tipo simple definido por el sistema o por el usuario. El dominio de un atributo define los valores posibles que puede tomar este atributo. Además de los dominios naturales, usados como tipos de datos. El administrador del sistema puede generar sus propios dominios definiendo el conjunto de valores permitidos. Esta característica usada en forma correcta, se convierte en mecanismo de control, restricción y validación de los datos a ingresar. (Rodríguez, 2011)

Reglas de integridad.

Integridad de entidad: Esta regla se aplica a las claves primarias de las relaciones base, ningún atributo que forme parte de una clave primaria puede aceptar valores nulos. Por definición, una clave primaria es irreducible y se utiliza para identificar de modo único los registros. Irreducible significa que ningún subconjunto de la clave primaria sirve para identificar las tuplas de modo único. Si se permite que parte de la clave primaria sea nula, se está diciendo que no todos sus atributos son necesarios para distinguir las tuplas, con lo que se contradice la irreductibilidad.

Esta regla sólo se aplica a las relaciones base y a las claves primaria, no a las claves foráneas ni alternativas. (Rodríguez, 2011)

Este tipo de integridad se cumple en las BD propuestas en ambas direcciones ya que cada tabla tiene definida su clave primaria. Las llaves primarias de las tablas son campos secuenciales o enteros que representan un código único, de esta forma este campo nunca será nulo ni se repetirá. Es de vital importancia especificar la clave principal para cada registro, ya que esta representa la herramienta fundamental que utiliza el servidor de BD para seleccionar la información que se necesite, evitando así la duplicidad de los registros.

Integridad de dominio: define valores válidos para los atributos de las diferentes tablas de una BD. Algunas tareas que aseguran la integridad de dominio son: definir los tipos de datos correctamente para cada campo y definir campos no nulos (NOT NULL) para evitar resultados inconsistentes. (Rodríguez, 2011)Para velar por la integridad de dominio en las BD, no se definieron atributos nulos y se precisaron correctamente cada uno de los tipos de datos para cada uno de los atributos.

Integridad referencial: define que la BD no debe contener valores de claves foráneas sin concordancia. Esta regla se aplica a las claves foráneas. Si en una relación hay alguna clave foránea, entonces sus valores deben coincidir con los valores de la clave primaria a la que hace referencia, o bien, debe ser completamente nulo. La integridad referencial implica que en todo momento todos los datos involucrados en la BD sean correctos y sin repeticiones innecesarias. Todas las BD relacionales presentan esta propiedad pues el software gestor es responsable de su cumplimiento. (Rodríguez, 2011)

La integridad referencial también vigila que se cumplan las siguientes reglas:

- No se podrá introducir un valor en la tabla relacionada si antes no ha sido introducida en la tabla principal.
- No se puede eliminar un registro de una tabla principal si existen registros coincidentes en la tabla relacionada.

- No se puede cambiar un valor de la clave primaria en la tabla principal si el registro tiene registros relacionados.

Análisis de la redundancia de la información.

La redundancia de datos es aquella información duplicada o almacenada varias veces en la misma base de datos. Esto dificulta la tarea de modificación de datos y es el motivo más frecuente de inconsistencia de los mismos. Además requiere un mayor espacio de almacenamiento, que influye en un mayor coste y tiempo de acceso a los datos.

Con un buen diseño de base de datos se logrará evitar la aparición de información repetida o redundante. Lo ideal es lograr una redundancia nula; no obstante, en algunos casos la complejidad de los cálculos hace necesaria la aparición de redundancias.

La base de datos para las direcciones de CITMA y Planificación Física se encuentra libre de redundancias, ya que se elimina casi completamente la presencia de datos repetidos innecesariamente.

3.4 Pruebas a la capa de acceso a datos.

Para verificar que los métodos implementados funcionan correctamente para los módulos de CITMA y Planificación Física se realizaron nueve Casos de Pruebas uno por cada método implementado GenericDAO.

A continuación se muestran las tablas donde se reflejan los aspectos que se tuvieron en cuenta para la realización de las pruebas a los métodos para la persistencia de objetos.

Método:	save	
Condiciones:	Debe existir el objeto de la entidad que se desee salvar.	
Objetivo de la Prueba	**Resultado Esperado**	**Resultado de la Prueba**
Insertar en la BD el objeto que fue pasado por parámetro.	Se espera que el objeto pasado por parámetro se almacene en la tabla correspondiente a la BD.	El objeto pasado por parámetro se almacenó satisfactoriamente.

Tabla 16. Prueba al Método save.

Método:	saveOrUpdate	
Condiciones:	Debe existir el objeto de la entidad que se desea salvar o actualizar.	
Objetivo de la Prueba	**Resultado Esperado**	**Resultado de la Prueba**
Insertar o actualizar en la BD el objeto que fue pasado por parámetro.	Se espera que de no existir los objetos en la base de datos sean guardados y si existen entonces actualizarlos en las tablas y columnas correspondientes en la base de datos.	Los objetos fueron actualizados correctamente en la base de datos.

Tabla 17. Prueba al Método saveOrUpdate.

Método:	find	
Condiciones:	Se le debe especificar la entidad que se desea buscar.	
Objetivo de la Prueba	**Resultado Esperado**	**Resultado de la Prueba**
Obtener el listado de la entidad especificada.	Se espera obtener una lista con todos los registros existentes en la entidad especificada.	Fue obtenido satisfactoriamente la lista con los registros de la entidad especificada.

Tabla 18. Prueba al Método find.

Método:	findByPK	
Condiciones:	Se le debe especificar la entidad y el valor de la llave primaria serializable.	
Objetivo de la Prueba	**Resultado Esperado**	**Resultado de la Prueba**
Obtener el objeto de la entidad especificada al que le corresponde la llave primaria indicada.	Se espera obtener el objeto que le corresponde la llave primaria indicada.	Fue obtenido el objeto especificado.

Tabla 19. Prueba al Método findByPK.

Método:	findByParamAndValue	
Condiciones:	Se le debe especificar la entidad, el parámetro y el valor por el que se desea buscar.	
Objetivo de la Prueba	**Resultado Esperado**	**Resultado de la Prueba**
Obtener el objeto de la entidad especificada que cumpla que el valor del parámetro se igual al especificado.	Se espera obtener el objeto cuyo valor del parámetro que se va a buscar sea igual al especificado.	Se obtuvo el objeto que presenta dicho parámetro con el valor indicado.

Tabla 20. Prueba al Método findByParamAndValue.

Método:	findByParamsAndValues	
Condiciones:	En la entidad debe existir dichos parametros	
Objetivo de la Prueba	**Resultado Esperado**	**Resultado de la Prueba**
Obtener el objeto de la entidad especificada que cumpla que el valor de los parámetros se igual a los especificados.	Se espera obtener el objeto cuyos valores de los parámetros que se va a buscar sean iguales a los especificados.	Se obtuvo el objeto que presenta los parámetros con los valores indicados.

Tabla 21. Prueba al Método findByParamsAndValues.

Método:	update	
Condiciones:	El objeto de la entidad que se desea actualizar debe existir.	
Objetivo de la Prueba	**Resultado Esperado**	**Resultado de la Prueba**
Actualizar los atributos del objeto de la entidad especificada.	Se espera que los atributos del objeto sean actualizados.	Los atributos del objeto fueron actualizados.

Tabla 22. Prueba al Método update.

Método:	delete	
Condiciones:	El objeto de la entidad especificada debe existir.	
Objetivo de la Prueba	**Resultado Esperado**	**Resultado de la Prueba**
Eliminar el objeto de la entidad especificada.	Se debe eliminar el objeto de la entidad.	El objeto de la entidad fue eliminado.

Tabla 23. Prueba al Método delete.

Método:	deleteAll	
Condiciones:	En la entidad deben existir al menos uno o varios objetos.	
Objetivo de la Prueba	**Resultado Esperado**	**Resultado de la Prueba**
Eliminar todos los objetos de la entidad especificada.	Se espera eliminar todos los objetos de la entidad.	Fueron eliminados los objetos de la entidad especificada.

Tabla 24. Prueba al Método deleteAll.

Conclusiones Parciales.

En este capítulo se analizaron los aspectos relacionados con la implementación y validación de la capa de acceso así como también la validación del diseño de la base de datos:

- Se describió el DAO Genérico así como los elementos necesarios para lograr la implementación de la capa de acceso a datos de los módulos de las direcciones del CITMA y Planificación Física.

- Se realizaron y mostraron las técnicas fundamentales para configurar Hibernate, y por consiguiente, lograr transacciones lógicas entre la aplicación y la base de datos.
- Se analizaron los aspectos a tener en cuenta para realizar el diseño de la BD, como son: la integridad de los datos y la redundancia de los mismos.
- Se realizaron los casos de pruebas a los métodos implementados en la capa de acceso a datos.

Conclusiones Generales.

Con el desarrollo del presente trabajo se realizó un estudio acerca de la manipulación de la información en las bases de datos así como de la persistencia y recuperación de los datos relacionales. En la misma fueron cumplidos los objetivos trazados:

- Se elaboró la Fundamentación Teórica de la Investigación.
- Se diseñó tanto la capa de acceso a datos como la base de datos para los módulos de las direcciones del CITMA y Planificación Física.
- Se implementó la capa de acceso a datos para los módulos de las direcciones del CITMA y Planificación Física.
- Se validaron los resultados obtenidos.

Recomendaciones

- Incluir nuevas funcionalidades para la persistencia de objetos.

Referencias Bibliográficas

García-Bustelo, Begoña Cristina Pelayo. 2007. *Tesis Doctoral. Desarrollo ágil de Softwarecon Arquitecturas Dirigidas por Modelos.* 2007.

A. Otto, D. Hinderink. 2004. *TYPO3 Database Abstraction.* 2004.

Army_Net_Centric_Data. 2009. Army Net-Centric Data Strategy. [En línea] 2009. http://data.army.mil/.

Campoy, Lourdes Arlín. 2009. Tutorial Base de Datos I. [En línea] 2009. http://sistemas.itlp.edu.mx/tutoriales/basedat1/tema1_1.htm..

Correa, Alejandra Patricia. 2010. Grupo de Investigación Aplicada para Estudiantes de Ing. de Sistemas de la UNEFA. [En línea] 2010. http://ing-de-sistemas-unefa.lacocte.

Gallego, Juan Pablo Gómez. 2007. RUP. [En línea] 27 de 9 de 2007. http://es.scribd.com/doc/297224/RUP.

Gil, Javier y Rosario. 2005. Sistema de Gestión de base de datos. [En línea] 2005. http://alfa.facyt.uc.edu.ve/computacion/pensum/cs0347/download/exposiciones/1/SGBD.pdf.

Hibernate.org. 2012. Hibernate. [En línea] 2012. http://www.hibernate.org/.

Infante, Daniel. 2009. Herramientas de Modelado. [En línea] 2009. http://www.geocities.ws/daniel.infante/fase2/t1.html.

Jahzeel. 2007. Cuadro Comparativo de Base de Datos. [En línea] 2 de 12 de 2007. http://jahzeel.espacioblog.com/post/2007/02/12/uadro-comparativo-bases-datos.

Mato, Rosa María García. 1999. *Diseño de Bases de Datos.* 1999.

Microsoft. 2011. Microsoft Open Database Connectivity (ODBC). [En línea] 2011. http://msdn.microsoft.com/en.../ms710252(v=vs.85).aspx..

Netbeans.org. 2012. Netbeans. [En línea] 2012. http://netbeans.org/index_es.html.

Nguèn, M. H. 2008. *Data Abstraction Layer: Independet for Database.* 2008.

Peñalver, Gladis. 2008. *MA-GMPR-UR2 Metodología ágil para proyectos de software libre: Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.* 2008.

Rodríguez, Yudirenia Torriente. 2011. *Capa de acceso a datos y modelo físico de la Base de Datos del Sistema Videoconferencia de la Facultad 6: Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.* 2011.

Rozic, Sergio Ezequiel. 2004. Base de Datos. [En línea] 2004. http://www.recol.es/index.php?option=com_content&task=view&id=110&Itemid=417

Schmieg, Sebastian. 2008. Fundamentación Informáticos. [En línea] 2008. http://fundamentosinformaticosjl.wordpress.com/category/base-de-datos/.

Bibliografía

García-Bustelo, Begoña Cristina Pelayo. 2007. *Tesis Doctoral. Desarrollo ágil de Softwarecon Arquitecturas Dirigidas por Modelos.* 2007.

A. Otto, D. Hinderink. 2004. *TYPO3 Database Abstraction.* 2004.

Army_Net_Centric_Data. 2009. Army Net-Centric Data Strategy. [Online] 2009. http://data.army.mil/.

Campoy, Lourdes Arlín. 2009. Tutorial Base de Datos I. [Online] 2009. http://sistemas.itlp.edu.mx/tutoriales/basedat1/tema1_1.htm..

Codd, Frank Edgar. 2009. Base de datos: Modelo de Datos - C# – ASP.NET – WEB – PHP. [Online] 2 6, 2009. http://www.imgeek.net/?p=542.

Correa, Alejandra Patricia. 2010. Grupo de Investigación Aplicada para Estudiantes de Ing. de Sistemas de la UNEFA. [Online] 2010. http://ing-de-sistemas-unefa.lacocte.

Cruz, Tavarez Carvalho. 2009. El modelo Entidad-Relación. [Online] 2009. http://civil.fe.up.pt/acruz/access/modeloER.htm..

Date. 2001. *Introduccion a las BD.* 2001.

Fernández, Luis H. 2009. Patrones de Diseño. [Online] 2009. http://software.guisho.com/patrones-de-diseno.

Gallego, Juan Pablo Gómez. 2007. RUP. [Online] 9 27, 2007. http://es.scribd.com/doc/297224/RUP.

Gil, Javier and Rosario. 2005. Sistema de Gestión de base de datos. [Online] 2005. http://alfa.facyt.uc.edu.ve/computacion/pensum/cs0347/download/exposiciones/1/SGBD.pdf.

Hibernate.org. 2012. Hibernate. [Online] 2012. http://www.hibernate.org/.

Infante, Daniel. 2009. Herramientas de Modelado. [Online] 2009. http://www.geocities.ws/daniel.infante/fase2/t1.html.

Jahzeel. 2007. Cuadro Comparativo de Base de Datos. [Online] 12 2, 2007. http://jahzeel.espacioblog.com/post/2007/02/12/uadro-comparativo-bases-datos.

Joshi, B. and Dickinson, P. 2001. *Professional ADO.NET.* 2001.

King, G. 2006. Hibernate Reference Documentation 3.2.2. [Online] 2006. http://www.hibernate.org/hib_docs/v3/reference/en/pdf/hibernate_reference.pdf.

Mato, Rosa María García. 1999. *Diseño de Bases de Datos.* 1999.

Microsoft. 2011. Microsoft Open Database Connectivity (ODBC). [Online] 2011. http://msdn.microsoft.com/en.../ms710252(v=vs.85).aspx..

Netbeans.org. 2012. Netbeans. [Online] 2012. http://netbeans.org/index_es.html.

Nguèn, M. H. 2008. *Data Abstraction Layer: Independet for Database.* 2008.

Pao, Angie. 2010. [Online] 2010. http://www.gratisblog.com/all_the_drama/i147406-base_de_datos_iv._modelos_de_datos.htm.

Parker, D. A. and Hoenicka, M. 2005. *Database Independent Abstraction Layer.* 2005.

Peñalver, Gladis. 2008. *MA-GMPR-UR2 Metodología ágil para proyectos de software libre: Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.* 2008.

Rodríguez, Rubén. 2004. Bases De Datos (La Historia). [Online] 1 2, 2004. http://www.fudim.org/comunicacion/notas/nota.php?id=22&a=Adim..

Rodríguez, Yudirenia Torriente. 2011. *Capa de acceso a datos y modelo físico de la Base de Datos del Sistema Videoconferencia de la Facultad 6: Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.* 2011.

Rozic, Sergio Ezequiel. 2004. Base de Datos. [Online] 2004. http://www.recol.es/index.php?option=com_content&task=view&id=110&Itemid=417

Sánchez, Fernando Luque. 2005. Aplicaciones en N - Capas usando Visual Basic .Net. [Online] 10 12, 2005. http://www.elguille.info/colabora/NET2005/FernandoLuque_NCapas.htm..

Schmieg, Sebastian. 2008. Fundamentación Informáticos. [Online] 2008. http://fundamentosinformaticosjl.wordpress.com/category/base-de-datos/.

Tentor, Julio. 2008. Arquitectura de N-Capas y N-Niveles. [Online] 2008. http://www.jtentor.com.ar/post/Arquitectura-de-N-Capas-y-N-Niveles.aspx..

Thelin, J. 2007. *Foundations of QT Development.* 2007.

Visconti, Marcello. 2008. *Fundamentos de Ingeniería de Software.* 2008.

Glosario de Términos.

BD: Base de datos.

SGBD: Sistema Gestor de Base de Datos. Es un conjunto de programas que permiten crear y mantener una Base de datos.

DSL: Sublenguaje de datos.

DDL: Lenguaje de definición de datos.

DML: Lenguaje de manipulación de datos.

DCL: Lenguaje de control de los datos.

API: Interfaz de Programación de Aplicaciones.

XML: Lenguaje de Marcas Extensible (Extensible Markup Language). Es un metalenguaje extensible de etiquetas. Permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

MVCC: Control de Concurrencia Multi-Versión.

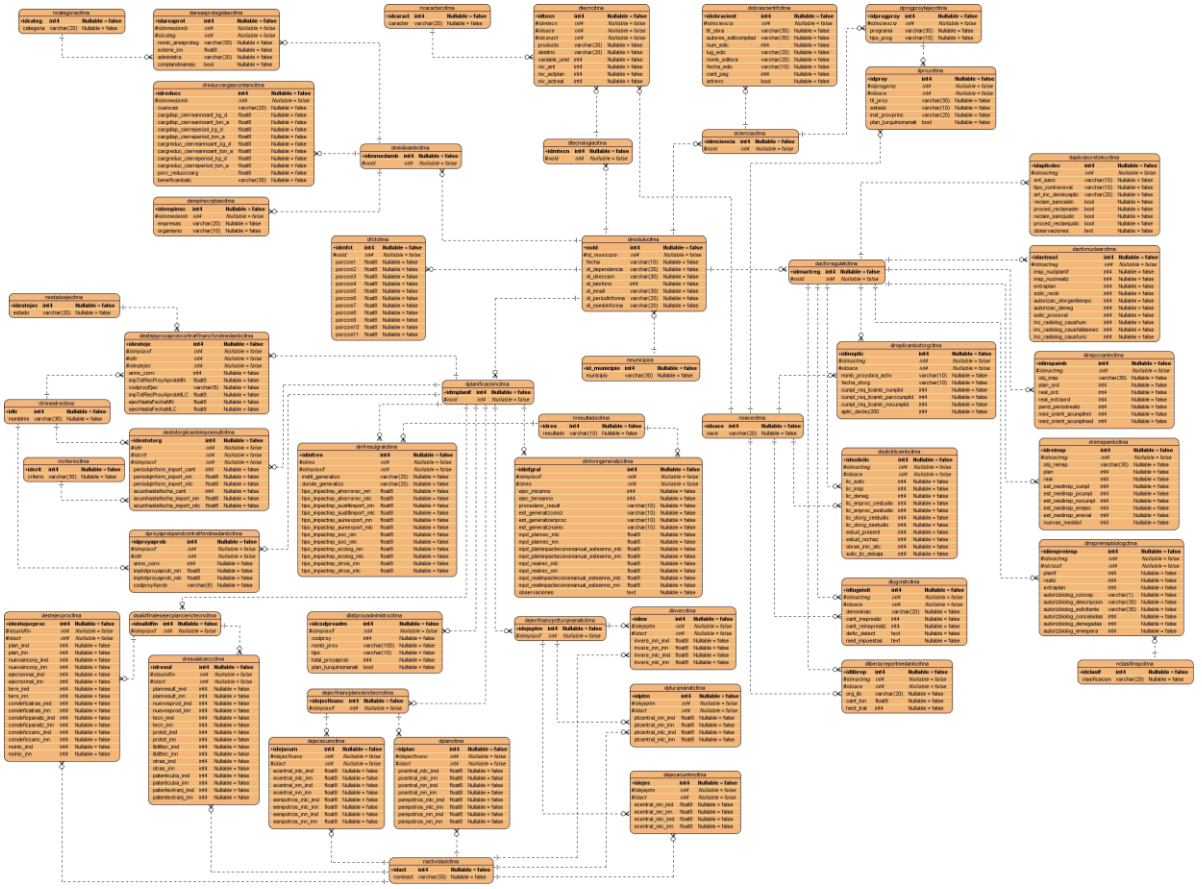
PHP: (Hypertext Preprocessor). Lenguaje de script diseñado para la creación de páginas web activas, multiplataforma.

HQL: Hibernate Query Language HQL. Lenguaje de consulta de Hibernate.

SQL: Structured Query Language. Es un lenguaje especializado de programación que permite realizar consultas

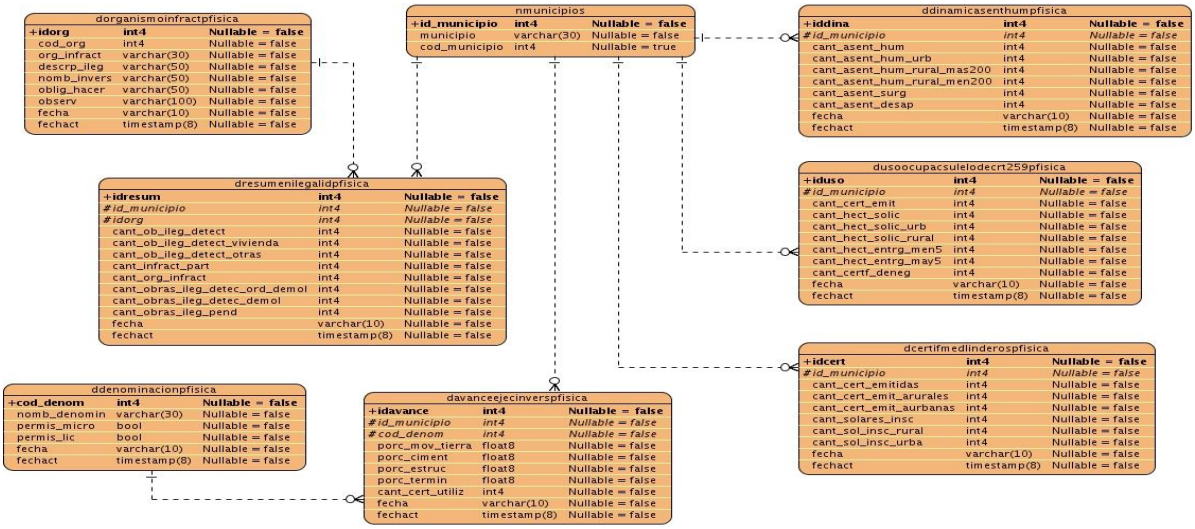
JDBC: Es el acrónimo de Java Database Connectivity, permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede utilizando el dialecto SQL del modelo de base de datos que se utilice.

Anexos.

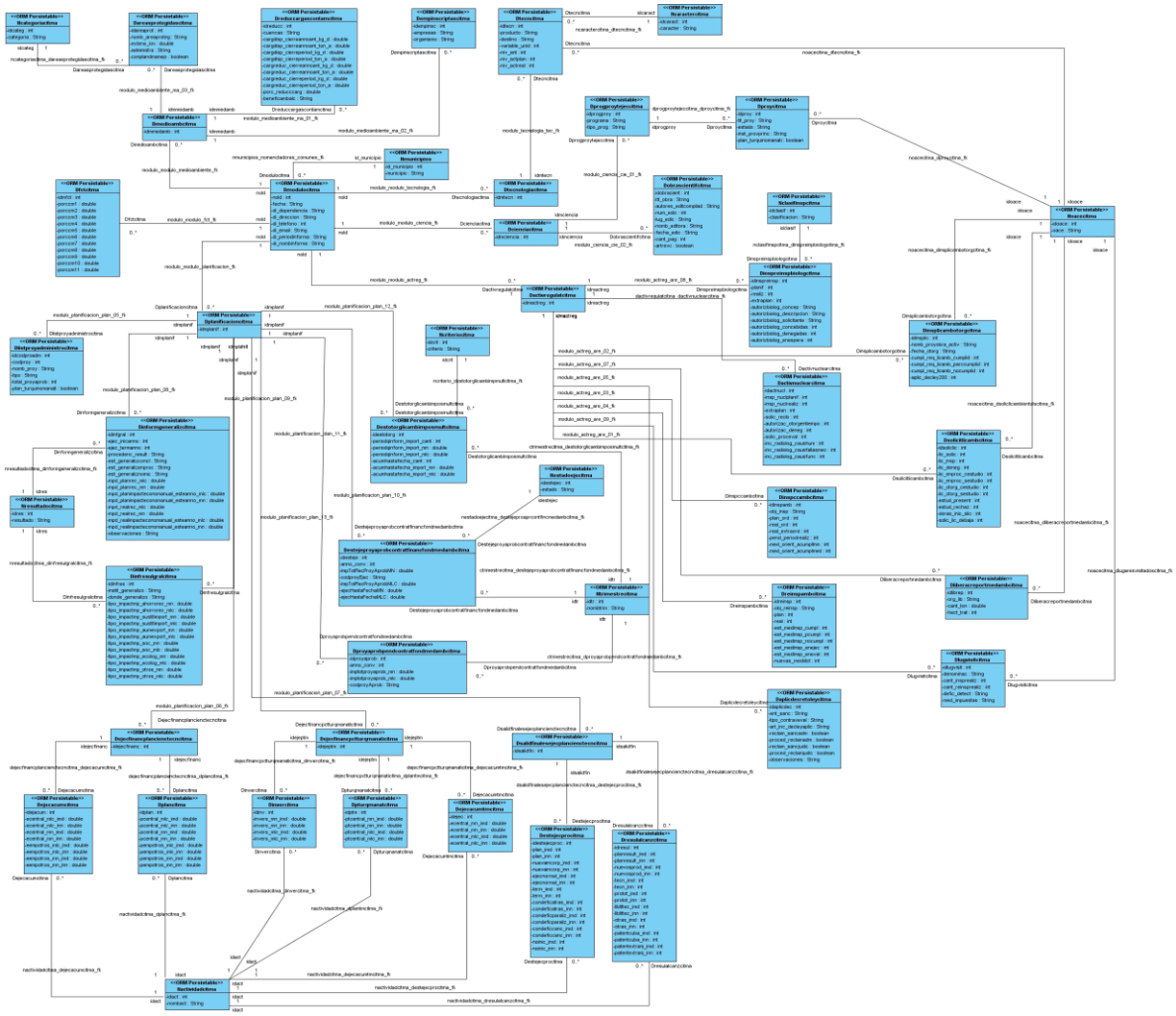


Anexo 1. DER

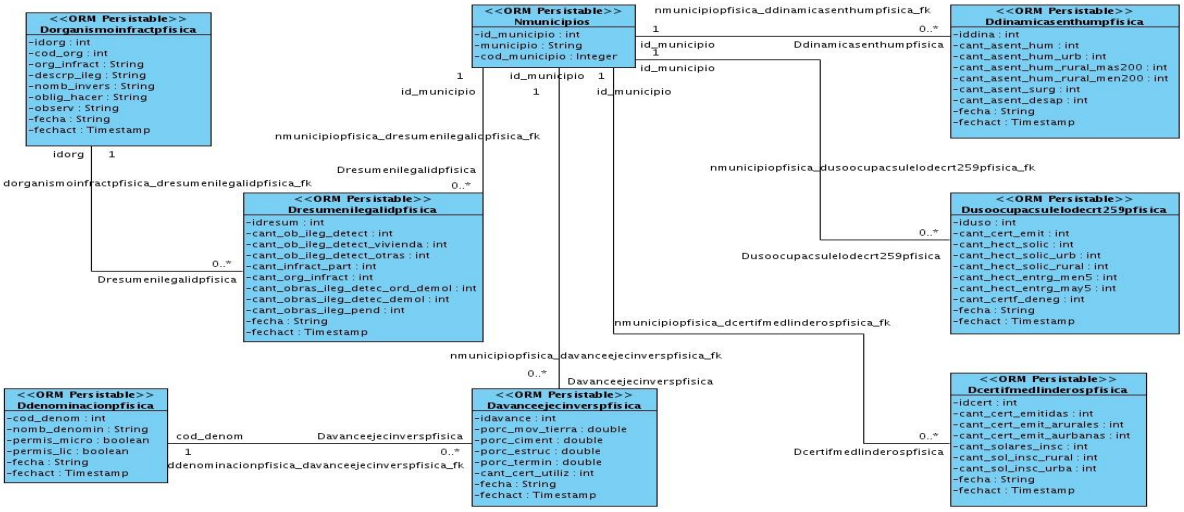
CITMA.



Anexo 2. DER Planificación Física.



Anexo 3. ORM CITMA.



Anexo 4. ORM Planificación Física.