Universidad de las Ciencias Informáticas

Facultad 7

Aplicación para la estandarización del modelo de datos del Sistema de Gestión para la Ingeniería Clínica y Electromedicina

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Autor: Miguel Ángel Arrebato Reyes

Tutor: Ing. Héctor Manuel Solís Mulet

La Habana, junio de 2011

"Año 53 de la Revolución"

DATOS DE CONTACTO

Tutor: Héctor Manuel Solís Mulet

Graduado con Título de Oro de Ingeniero en Ciencias Informáticas en la Universidad de la Ciencias Informáticas (UCI) en el año 2008. Posee Categoría Docente de Instructor. Ha participado en proyectos de desarrollo de Sistemas Informáticos para la Salud desde el año 2005. Imparte la asignatura de Gestión de Software y Sistemas de BD. Actualmente se desempeña como Jefe de Colectivo de las asignaturas que imparte en la Facultad 7 y es miembro del grupo de BD del Departamento de Sistemas de Apoyo a la Salud (SAS) del Centro Especializado en Soluciones Informáticas Médicas (CESIM).

Correo electrónico: 1

Agradecimientos

Son tantas las personas que quiero agradecer y tantos nombres que no puedo olvidar porque me sentiría mal si se me quedara alguien por eso pido disculpas este seguro que esta en mi corazón mis mas sincero cariño y agradecimiento. A partir de estas líneas que dejan huellas indelebles para la historia queda plasmada mi gratitud hacia todos aquellos que de una manera u otra contribuyeron a la realización de este trabajo de diploma.

- *A la Universidad de las Ciencias Informáticas y la Revolución Cubana por darme la oportunidad gratuita e incondicional de prepararme como ingeniero durante todo el transcurso de la carrera.*

- *A mi tutor el ingeniero Héctor Manuel Solís Mulet por darme todo el apoyo que necesitaba y por saber guiarme hacia la mejor solución de los problemas, por ser paciente y amigo gracias. no hubiese querido tutor mejor que tú.*

- *A todos los profesores del Departamento Sistemas de Apoyo a la Salud y a los profesores de Electro medicina la ingeniera Velmur Muñoz Casals que siempre nos defiende a capa y espada, a la ingeniera Soyla, al ingeniro Rannier Rivero Sevila, al ingeniero Arieskien Mendoza Guerra mi respeto para ti, no falto un día que no mostrara preocupación por el trabajo, eso no lo voy a olvidar nunca, al ingeniero Dennys Javier Hernández Peña y a la ingeniera Yislenis Suarez Hernández, gracias .*
- *A mi familia que no es muy grande, a mi mama Mabel Reyes Guía por todo su amor y cariño, por siempre estar ahí en los momentos difíciles dándome fuerzas para seguir adelante cuando pensé que no podía, te amo mami, gracias, a mi hermano Enmanuel Reyes Guía que siempre esta conmigo en todo momento y que comprende mis palabras mejor que nadie , te quiero con la vida, a mis tías Mayra Reyes Guía, Caridad y Mercedes por siempre brindarme su amor y cariño por siempre estar ahí para mi desde que naci. A Emilio gracias por todo el apoyo que nos as brindado, no te imaginas cuanto te lo agradezco. A mi padre Gustavo Julián Arrebato Guillen y a mi tío Orlando Arrebato Guillen por guiarme por el buen camino a mis hermanos Alain y Gustavito por estar presente y pendiente de mi a pesar de la distancia.*

- *A mis amigos y compañeros del barrio a raudel, lumey, dazi, yamile, Irene, Ronco, Héctor, Mirta, Juana, Guillermo a los demás vecinos del barrio, a mis amigos y compañeros de la universidad que siempre me han*

apoyado y alentado para seguir adelante en especial el grupo 7503 y a todos los del edificio 89. Gracias por dejarme compartir con ustedes todos estos años, recuerdos que me llevo para toda la vida.

Dedicatoria

Quiero dedicar este Trabajo de Diploma a mi madre Mabel Reyes Guía, por haber logrado parte de mis sueños, por todo el sacrificio que a tenido que hacer para que hoy su hijo pueda estar donde esta, por ser la base de mi existencia, por no darse por vencida, sin la ayuda de nadie logra grandes cosas en nuestras vidas, por ser madre y padre a la vez, por servirme siempre de ejemplo. Te quiero mami, esto se logro gracias a ti.

RESUMEN

El Centro Nacional de Electromedicina (CNE) ha promovido el desarrollo de un sistema informático para llevar el control y mantenimiento de equipos médicos a nivel nacional, para darle solución a este objetivo se encuentra en desarrollo el Sistema de Gestión para la Ingeniería Clínica y Electromedicina (SIGICEM). La gestión de toda esta información es un proceso complicado y se hace aún más complejo al no contar con un modelo que le permita exportar su base de datos (BD) hacia otros Sistemas de Gestión de Base de Datos (SGBD), cuestión que les dificulta la labor de migración de su extenso modelo de datos hacia otros SGBD.

El presente trabajo propone solucionar este problema a través de una aplicación informática que permita estandarizar el modelo de datos del SIGICEM, característica que le permitirá exportar su modelo de datos hacia otros gestores de BD. Para ello se emplearán las herramientas (DBdesigner Fork y el Framework Symfony) más convenientes para la Universidad de las Ciencias Informáticas (UCI) relacionadas con el almacenamiento de la información así como las últimas tendencias en el mundo de la Informática. Este trabajo permitirá migrar el modelo de datos del sistema hacia distintos gestores de base de datos y agilizar el desarrollo de los sistemas que utilicen la aplicación.

Palabras Claves: estandarizar, base de datos.

TABLA DE CONTENIDO

Introducción

Los sistemas de información existen desde las primeras civilizaciones. Los datos se recopilaban, se estructuraban, se centralizaban y se almacenaban convenientemente. El objetivo inmediato de este proceso era poder recuperar estos mismos datos u otros derivados de ellos en cualquier momento, sin necesidad de volverlos a recopilar, este paso solía ser el más trabajoso trayendo consigo deficiencia en el resultado de la búsqueda. Desde la antigüedad ha sido necesario almacenar grandes cantidades de datos con el objetivo de tomar decisiones y realizar acciones más pertinentes que las que se realizarían sin dicha información, es por ello que surge lo que se conoce como BD. (1)

El surgimiento de las BD estuvo motivado por la necesidad de las grandes empresas de almacenar amplias cantidades de información de una forma rápida, sencilla y fiable, sin necesidad de desplazarse a salas dedicadas a archivar documentación. Este es el lugar donde se agrupa un conjunto de datos organizados que pertenecen a un mismo contexto, con el objetivo de ser posteriormente consultados. En este sentido, una biblioteca puede considerarse una BD compuesta en su mayoría por documentos indexados para su consulta. En la actualidad, debido al desarrollo de campos como la informática y la electrónica, la mayoría de estas tienen formato electrónico, que ofrece un amplio rango de soluciones al problema de almacenar datos.

Con este avance aparecen nuevos dispositivos electromagnéticos que ofrecen mayores posibilidades de almacenamiento y economizan un tiempo considerable en la búsqueda y tratamientos de datos, esto se ha logrado mediante el uso de diferentes metodologías de diseño, herramientas como los SGBD y las aplicaciones que la utilizan característica que permite el desarrollo de aplicaciones que facilitan la comunicación y la gestión de grandes volúmenes de datos. (2)

Tras el triunfo de la revolución cubana, el 1ro de agosto de 1961 se funda el Ministerio de Salud Pública (MINSAP) y se concibe la construcción de nuevos policlínicos, hospitales y pequeñas unidades de salud, que se dotaron con las nuevas tecnologías. Para lograr el objetivo de obtener un mejor funcionamiento de los equipos médicos, debido al desarrollo en las unidades de salud se hacía necesario crear talleres que proporcionaran la reparación y mantenimiento de los mismos. Por lo que el Centro Nacional de Electromedicina (CNE) se proyecta con una nueva misión: garantizar la sostenibilidad de la tecnología

médica mediante el correcto asesoramiento de las adquisiciones de equipos y piezas, y la prestación de servicios técnicos de excelencia durante el tiempo de explotación de cada tecnología. El MINSAP impulsado por la dirección del país se propuso informatizar su sector para facilitar el intercambio de información entre los especialistas del país.

A partir del año 2008 después de haberse reestructurado la misión y el alcance del CNE, se decide cambiar el nombre actual del centro por uno que respondiera claramente a la nueva actividad que se realizaría, la cual vincularía la ingeniería clínica con la electromedicina, por lo que se le denominó Centro de Ingeniería Clínica y Electromedicina (CICEM).

Actualmente se encuentra en desarrollo el Sistema de Gestión para la Ingeniería Clínica y Electromedicina (SIGICEM), el mismo es realizado en la Universidad de las Ciencias Informática por los desarrolladores del Departamento Sistemas de Apoyo a la Salud. Los desarrolladores del mismo utilizan el Framework Symfony como entorno de desarrollo ya que este simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Este framework también posee un mecanismo que permite que su modelo de datos pueda ser exportado en cualquier momento del desarrollo del software a diferentes sistemas de almacenamientos de datos, lo cual resulta una característica muy atractiva pues evita gran parte de las dificultades que surgen una vez que es necesario migrar de un SGBD a otro.

La utilización de tecnologías representa un cambio sustancial en la manera en que se obtiene el modelo de datos de un software, debido a que este exige una estructura predefinida a fin de poder ofrecer beneficios como los antes mencionados. Los desarrolladores del SIGICEM no cuentan con una herramienta que les permita de manera automatizada obtener modelos de datos estándar para el Framework Symfony, es un proceso que actualmente se lleva a cabo de forma manual y de manera específica para cada sistema informático en desarrollo.

Existen intentos con el propósito para resolver la problemática que representa la obtención del modelo de datos estándar para este framework, un ejemplo de ello lo constituye el Trabajo de Colaboración Médica, desarrollado en la UCI. Con la realización del mismo sus desarrolladores se propusieron obtener un modelo de datos compatible al requerido por el Framework Symfony, esto se logró aunque de manera muy particular, solo para resolver las necesidades de la información propia de esta aplicación informática.

Luego de un análisis de la problemática anterior y con el fin de solucionar estas necesidades se propone como **problema a resolver:** ¿Cómo gestionar la estandarización del modelo de datos de SIGICEM?

Para enmarcar los límites de esta investigación se define como **objeto de estudio:** Proceso de estandarización de modelos de datos para el Framework Symfony, delimitando el **campo de acción en:** Proceso de estandarización del modelos de datos de SIGICEM.

La presente investigación tiene como **objetivo general:** Desarrollar una aplicación que gestione la estandarización del modelo de datos de SIGICEM.

Para dar cumplimiento a lo anteriormente planteado, se trazaron las siguientes **tareas de investigación:**

1. Realizar un estudio del arte referente a la estandarización de modelos de datos.
2. Aplicar los lenguajes de programación y tecnologías necesarias para la implementación de la solución propuesta: PHP, XML, XSL, YML, ORM, Propel, Doctrine.
3. Personalizar un documento XSL propuesto que realiza transformaciones que tienen como resultado un YML optimizado para Propel en función de las pautas de diseño de BD utilizadas por los desarrolladores de SIGICEM.
4. Implementar una aplicación que permita transformar un documento XML dado en un esquema YML optimizado para Propel.
5. Validar la solución propuesta para el ORM Propel.
6. Entregar la documentación de la solución propuesta.

Con la realización de este trabajo se obtendrá como principal beneficio una aplicación que generará modelos de datos compatibles con el Framework Symfony y a la vez acorde con las pautas de diseño de bases de datos utilizadas por los desarrolladores del SIGICEM. Esto permitirá disfrutar de todos los beneficios que brinda Symfony relacionados con la capa de datos, brindará de manera especial la facilidad de migración de un SGBD a otro. La utilización de esta aplicación permitirá estandarizar modelos de datos provenientes de desarrolladores de otros proyectos que utilicen este framework de desarrollo de software.

La investigación consta de 3 capítulos, en los cuales se desarrollan aspectos de importancia para lograr el objetivo que se persigue.

Capítulo #1: Fundamentación teórica

En el presente capítulo se brinda información acerca del estudio realizado sobre los procesos de estandarización de modelos de datos para el Framework Symfony, técnicas empleadas para su diseño y modelación, las herramientas y valorar cuál es el modelo.

Capítulo #2: Análisis y descripción de la solución propuesta

El presente capítulo describe, la estrategia de integración con el Framework Symfony, la descripción de la arquitectura a utilizar, una breve descripción de las tablas más significativas los valores que almacenan las mismas y la arquitectura de la propuesta de solución.

Capítulo #3: Implementación y pruebas

En este capítulo se ofrece información sobre la validación teórica y funcional del modelo de datos propuesto por el SIGICEM. Se detallan los elementos necesarios para la creación de un plugin para Symfony y se analiza la solución propuesta así como las pruebas realizadas a la misma.

Capítulo1: Fundamentación Teórica

En el presente capítulo se brinda información acerca del estudio realizado sobre los procesos de estandarización de modelos de datos para el Framework Symfony, técnicas empleadas para su diseño y modelación, las herramientas y valorar cuál es el modelo.

1.1. Base de Datos

Es un componente fundamental de un sistema de información. El escenario actual de la tecnología de BD es resultado del perfeccionamiento que ha tenido lugar en el procesamiento de datos y en la gestión de la información, conjunto de datos que pertenecen al mismo contexto almacenados sistemáticamente para su posterior uso. En este sentido, una biblioteca puede considerarse una BD compuesta en su mayoría por documentos y textos impresos en papel e indexados para su consulta. Debido al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría de las bases de datos tienen formato electrónico, lo que ofrece un amplio rango de soluciones al problema de almacenar datos, por lo cual se hace necesario que sus características fundamentales queden recogidas en esta primera parte investigativa.

Tipología de Base de Datos

Estas pueden ser clasificadas desde diferentes puntos de vista en cuanto a diversos criterios de clasificación como son:

- ☑ **La variabilidad de los datos almacenados**

BD estáticas: Son de sólo lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones y tomar decisiones. Estas que no se puede modificar la información que está guardada, es decir, solo se pueden consultar. (3)

BD dinámicas: la información almacenada se modifica con el tiempo, por lo que en ellas están presentes todas las operaciones de gestión (agregar, modificar-actualizar y eliminar) por lo que la información guardada en ellas sufre cambios constantemente. (4)

- ☑ **Modelo de administración de datos**

Modelo relacional: Organiza la información en forma de tablas bidimensionales, de tal forma que estas son percibidas por los usuarios de igual modo. Para interactuar con la información en dichas tablas este modelo tiene definido un grupo de operadores, y para construir las consultas a BD relacionales el lenguaje más utilizado es el Lenguaje Estructurado de Consultas (SQL), un estándar implementado por los principales motores. Durante su diseño se pasa por un proceso conocido como normalización, con el fin de eliminar redundancias y evitar inconsistencias en las tablas.

Modelo orientado a objetos: Conjuga de forma centralizada los conceptos de abstracción, jerarquía, modularidad, persistencia, tipos y concurrencia. Un modelo orientado a objetos tiene obligatoriamente que cumplir con lo siguiente: la estructura básica de trabajo son los objetos, no algoritmos; donde cada objeto no es más que una instancia de una clase ya definida y que dichas clases estarán relacionadas únicamente por relaciones de herencia.

Una BD orientada a objetos incorpora todos los conceptos importantes del paradigma de objetos:

- ☑ Encapsulación - Propiedad que permite ocultar la información al resto de los objetos, característica que impide así accesos incorrectos o conflictos.
- ☑ Herencia - Propiedad a través de la cual los objetos heredan comportamiento dentro de una jerarquía de clases.
- ☑ Polimorfismo - Propiedad de una operación mediante la cual puede ser aplicada a distintos tipos de objetos.

En BD orientadas a objetos, los usuarios pueden definir operaciones sobre los datos como parte de la definición de la BD. Una operación (llamada función) se especifica en dos partes. La interfaz (o signatura) de una operación incluye el nombre de la operación y los tipos de datos de sus argumentos (o parámetros). La implementación (o método) de la operación se especifica separadamente y puede modificarse sin afectar la interfaz. Los programas de aplicación de los usuarios pueden operar sobre los datos ejecutados a dichas operaciones a través de sus nombres y argumentos, sea cual sea la forma en la que se han implementado. Esto podría denominarse independencia entre programas y operaciones.

Modelo jerárquico: Este modelo utiliza árboles para la representación lógica de los datos. Este árbol está compuesto de elementos llamados nodos. El nivel más alto del árbol se denomina raíz. Cada nodo

representa un registro con sus correspondientes campos. Son especialmente útiles en el caso de aplicaciones que manejan un gran volumen de información y datos muy compartidos lo cual permite crear estructuras estables y de gran rendimiento. Una de las principales limitaciones de este modelo es su incapacidad de representar eficientemente la redundancia de datos. (5)

Modelo de red: En este modelo las entidades se representan como nodos y sus relaciones son las líneas que los unen. En esta estructura cualquier componente puede relacionarse con cualquier otro. (6)
Los conceptos básicos en el modelo en red son:

- ☑ Un hijo puede tener varios padres.
- ☑ El tipo de registro, que representa un nodo.
- ☑ Elemento, que es un campo de datos.
- ☑ Agregado de datos, que define un conjunto de datos con nombre.

Elección para la solución en propuesta

El modelo de datos del SIGICEM sufrirá cambios constantemente, por lo que la misma será de tipo dinámica. Actualmente la BD del SIGICEM es de modelo relacional, la misma es la más factible para aplicarle la solución propuesta, aunque existen otras tipologías como la orientada a objetos pero estas imponen restricciones como la estructura básica (los objetos). Además, tipologías como la jerárquica o de modelo de red son de muy complejo entendimiento y sus mecanismos de reducción de redundancia son deficientes. Por otra parte, el modelo relacional brinda las siguientes ventajas:

- ☑ Fácil comprensión: Al ignorar el almacenamiento físico de los datos, se centra en el modelo lógico de la BD, por lo que su interpretación humana resulta más sencilla.
- ☑ Garantiza la integridad referencial: Al eliminar un registro elimina todos los registros relacionados dependientes.
- ☑ Posee mecanismos de reducción de redundancia de datos: Estos mecanismos permiten optimizar y evitar inconsistencias o duplicidad de datos.

1.2. Antecedentes

Existen intentos con el objetivo de resolver la problemática que representa la obtención del modelo de datos estándar para el Framework Symfony, un ejemplo de ello lo constituye:

La aplicación sfDB4toPropelPlugin, plugin desarrollado para el Framework Symfony con el objetivo de obtener un modelo de datos compatible este framework. Para que este funcione es necesario la utilización de la herramienta DBdesigner Fork v1.0 utilizada para el diseño del modelo de datos. Ésta aplicación se tomó como base de estudio para la implementación de un componente que logrará estandarizar el modelo de datos del SIGICEM para el Framework Symfony, pues mediante este plugin no se logró estandarizar el modelo de datos del SIGICEM, debido a la poca programación, documentación y configuración insuficiente que presenta esta aplicación para lograr estandarizar el modelo de datos del SIGICEM.

Otro ejemplo lo constituye el trabajo de Colaboración Médica, desarrollado en la UCI. Con la realización del mismo sus desarrolladores se propusieron obtener un modelo de datos compatible al requerido por el Framework Symfony, este fue implementado en el sistema de almacenamiento de datos PostgreSQLv8.3. Se utilizó como herramienta de modelado el Enterprise Architect v7.1 y para la implementación del modelo de datos, el Mapeo de Objeto Relacional (ORM) Doctrine del Framework Symfony.

Los objetivos que se esperaban de este trabajo fueron cumplidos, pues se logró estandarizar el modelo de datos del sistema de colaboración médica. Pero la propuesta analizada no cumple con las necesidades del SIGICEM, pues se realizó de forma manual y específicamente para el sistema de Colaboración Médica, solo fue para resolver las necesidades de información propias de esta aplicación informática. (5)

1.3. Tendencias y tecnologías actuales

Para lograr un buen diseño de un modelo de datos se necesita de una serie de herramientas que permitan la gestión de los procesos inherentes, control sobre la redundancia de los datos, así como su consistencia, comparación de archivos por usuarios autorizados.

La integridad de la documentación, la seguridad con protección frente a usuarios no autorizados, la accesibilidad de la información a través de los SGBD, mantenimiento e independencia de los datos, concurrencia, servicios de copias de seguridad y de recuperación ante fallos, así como, lograr que todo el proceso de creación y modelado sea lo más sencillo y rápido posible. A continuación se exponen características de herramientas que facilitan el proceso de diseño y modelado de datos.

1.4. Herramientas de desarrollo

- ☑ **DB Designer Fork 1.0**

Programa de diseño visual de modelos de datos que integra el diseño entidad-relación con la creación de BD. Una vez creado el modelo de datos, se podrán establecer los scripts correspondientes para diferentes gestores, como Firebird/InterBase, Microsoft SQL Server, MySQL, Oracle o PostgreSQL.es un sistema totalmente visual de diseño de bases de datos, que combina características y funciones profesionales con un diseño simple, muy clara y fácil de usar, a fin de ofrecerte un método efectivo para gestionar tus bases de datos. Permite administrar la base de datos, diseñar tablas, hacer peticiones SQL manuales y mucho más, como ingeniería inversa en MySQL, Oracle, MSSQL y otras bases de datos ODBC y modelos XML. (7)

- ☑ **NetBeans 6.9**

El Entorno Integrado de Desarrollo (IDE) NetBeans v6.9 es una herramienta pensada para escribir, compilar, depurar y ejecutar programas. Consta de una gran comunidad de usuarios en constante crecimiento, lo que le ha permitido, al igual que a muchos otros sistemas libres, el progreso paulatino de sus prestaciones y la eliminación de errores que pudiesen existir. El soporte para PHP ha sido extendido para incluir el Framework Symfony y PHP 5.3.

Es gratuito para desarrolladores de software y ofrece todas las herramientas necesarias para crear aplicaciones Web y de escritorio con el lenguaje Java, C/C ++ y lenguajes dinámicos como PHP y Java Script. Es fácil de instalar y se puede ejecutar tanto en Windows como en Linux; también tiene detección de errores de sintaxis en tiempo real. Permite que las aplicaciones se desarrollen a partir de un conjunto de módulos o componentes de software. Brinda una barra de navegación para el acceso rápido a funciones en una clase muy extensa, además de un completamiento de código fuente eficiente y seguro. (8)

- ☑ **Apache JMeter**

JMeter una herramienta Java dentro del proyecto de Jakarta que permite realizar pruebas de rendimiento y pruebas funcionales sobre aplicaciones web y BD. El proyecto Jakarta es el que se encarga de crear y mantener todas las soluciones Open Source (código abierto) creadas para la plataforma Java. Se destaca por su versatilidad, estabilidad, y por ser de uso gratuito. Posibilita que se ejecuten pruebas webs clásicas, test de FTP, JDBC, Web Service (en Beta) entre otras. Permite la ejecución de pruebas distribuidas entre

distintos ordenadores para realizar pruebas de rendimiento además de mostrar los resultados de las pruebas en una amplia variedad de informes y gráficas. (9)

Con Apache JMeter se puede llevar un control de las aplicaciones web y del servidor que la hospeda. Esto viene muy bien ya que muchas veces, no se puede controlar si soportará tanto público. Mediante este se puede realizar pruebas de parámetros de sesión, ítems, acciones y resultados. Además permite otras opciones como el acceso a los servicios FTP, HTTP, conexiones TCP o a bases de datos.

- ☑ **Data Generator**

Los desarrolladores y administradores conocen deben conocer la rutina de pruebas. Una de las tareas más populares es la de crear conjunto de datos para nuevas BD.

En promedio, toma desde varias horas hasta varios días de trabajo completo, cada dato tiene que ser creado manualmente, luego diferentes parámetros deben ser cambiados uno por uno para asegurarse que todo funcione correctamente.

El Data Generator es un utilitario simple, poderoso y completamente ajustable que genera datos para realizar pruebas a BD. La gran ventaja de este programa es que es capaz de crear una amplia variedad de tablas de prueba y soporta plantillas definidas por el usuario. Estos datos pueden ser fácilmente insertados. Hay varias formas de llenarlas. Aleatoriamente, por máscara o de datos de otras tablas.

Es importante mencionar que cada tabla se controla separadamente. Un paquete de datos puede contener múltiples tablas, cada uno con sus propias columnas, rango de valores y parámetros, lo cual hace que el uso de la herramienta sea muy conveniente. Adicionalmente, el programa es capaz de crear sentencias SQL para cada una de las operaciones. Más aun, este Data Generator resuelve la integridad de las llaves foráneas y el orden de las tablas es de forma automática.

Es muy efectivo y trabaja rápido. Es la herramienta a poseer por cada desarrollador o administrador quien desea ahorrar tiempo. Data Generator da la opción de realizar las comprobaciones en distintos servidores mediante ODBC. (10)

1.5. Tecnologías del software

- ☑ **Framework Symfony 1.4**

Diseñado con el objetivo de optimizar la creación de las aplicaciones web con el uso de sus características. Posee una librería de clases que permiten reducir el tiempo de desarrollo de las aplicaciones web. Está desarrollado en PHP5, se puede utilizar en plataformas como: Unix, Linux y Windows. Requiere de una instalación, configuración y líneas de instrucción, incorpora el patrón Modelo Vista Controlador (MVC), el modelo representa la información con la que trabaja la aplicación y se encarga de acceder a los datos. La vista transforma la información obtenida por el modelo en las páginas web a las que acceden los usuarios. El controlador es el encargado de coordinar todos los demás elementos y transformar las peticiones del usuario en operaciones sobre el modelo y la vista.

Es un enorme conjunto de herramientas y utilidades que minimizará el desarrollo de la aplicación a desarrollar, ya que es una de las mejores copias para PHP del famoso Framework Ruby on Rails. También ha tomado las mejores ideas de Rails y de muchos otros framework, características que le permiten añadir ideas propias y el resultado es un framework elegante, estable, productivo y muy bien documentado.

Entre las múltiples ventajas técnicas se pueden citar:

- ☑ **Abstracción de BD vía Mapeo Relacional de Objetos (ORM):** Las tablas están disponibles como objetos en el código. La capa ORM está basada en Propel o Doctrine.
- ☑ **Un parseador YML (YAML):** Propio, de forma que los ficheros de configuración y la descripción del modelo de datos pueden ser descritos de forma sencilla y rápida (a diferencia de los ficheros XML, con una gran cantidad de tags de apertura y cierre). Documentación de gran calidad, así como una amplia y activa comunidad de desarrolladores.
- ☑ **Genera código orientado a objetos**: Para las funcionalidades más comunes del manejo de BD.
- ☑ **Genera interfaces CRUD (Create, Read, Update and Delete):** Para las tablas.
- ☑ **Fácil de instalar y configurar:** en sistemas Windows, Mac y Linux.
- ☑ **Funciona con todas las bases de datos comunes:** (MySQL, PostgreSQL, SQLite, Oracle, MS SQL Server). (11)
- ☑ **Mapeo Relacional de Objetos (ORM)**

Un ORM es una componente de software que permite trabajar con los datos persistidos como si ellos fueran parte de una base de datos orientada a objetos. Debido a que lo estándar es trabajar con BD relacionales, se deben realizar operaciones que permitan transformar un registro en objeto y viceversa, a esta funcionalidad se la llama Mapeo objeto-relacional, es una técnica de programación que permite convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y es utilizado en una BD relacional, es decir, las tablas pasan a ser clases y los registros, objetos que se pueden manejar con facilidad. Utilizar un ORM tiene una serie de ventajas que facilitan enormemente tareas comunes y de mantenimiento:

- ☑ **Reutilización:** La principal ventaja que aporta un ORM es la reutilización del código desarrollado, característica que permite llamar a los métodos de un objeto de datos desde distintas partes de la aplicación e incluso desde diferentes aplicaciones.
- ☑ **Encapsulación:** La capa ORM encapsula la lógica de los datos, la misma consigue hacer cambios que afectan a toda la aplicación únicamente al modificar una función.
- ☑ **Portabilidad:** Utilizar una capa de abstracción permite cambiar en mitad de un proyecto de ejecutado en el sistema MySQL a cualquier SGBD sin ningún tipo de complicación. Esto es debido a que no utilizan una sintaxis específica de un SGBD para acceder al modelo, sino una sintaxis propia del ORM utilizado que es capaz de traducir a diferentes tipos de BD.
- ☑ **Seguridad:** Los ORM suelen implementar mecanismos de seguridad que protegen la aplicación de los ataques más comunes como inyecciones SQL.
- ☑ **Mantenimiento del código:** Gracias a la correcta ordenación de la capa de datos, modificar y mantener el código es una tarea sencilla. (12)

Doctrine y Propel son tipos de ORM para PHP 5.2.3 y posteriores versiones que se sitúan encima de una potente capa de abstracción de BD. Una de sus principales características es la opción de escribir las consultas en un lenguaje personalizado. Esto proporciona a los desarrolladores una poderosa alternativa a SQL que mantiene la flexibilidad, sin necesidad de la duplicación de código innecesaria.

- ☑ **ORM Doctrine y Propel**

Existen diversos ORM utilizados por desarrolladores de aplicaciones web, entre los mas utilizados se encuentran Doctrine y Propel**:**

Doctrine

Doctrine es una librería para PHP que permite trabajar con un esquema de BD como si fuese un conjunto de objetos, y no de tablas y registros. Está inspirado en Hibernate, que es uno de los ORM más populares y grandes que existen además de brindar una capa de abstracción de la BD muy completa. La característica más importante es que da la posibilidad de escribir consultas de BD en un lenguaje propio llamado Doctrine Query Language (DQL).Entre las múltiples ventajas técnicas de Doctrine, se pueden citar:

- ☑ **Generación automática del modelo**: El trabajo con ORM, genera un conjunto de clases que representa el modelo de la aplicación, Doctrine genera de forma automática el modelo de clases basándose en el modelo relacional de tablas. Es decir, si se tiene una tabla llamada usuarios, se autogenerará una clase llamada Usuarios cuyas propiedades son las columnas de dicha tabla.
- ☑ **Posibilidad de trabajar con YAML:** Doctrine puede generar de forma automática el modelo, pero también deja la posibilidad (como es lógico) que se pueda definir el mapeo de tablas y sus relaciones. Esto se puede hacer con código PHP o con YAML, que es un formato de datos legible muy usado por humanos para este fin. **(13)**

Propel

Propel utiliza PHP Data Objects *(*PDO) como capa de abstracción, y la generación de código para eliminar la carga de la inspección interna en tiempo de ejecución. Por lo tanto es rápido. No es necesario preocuparse por las conexiones de la BD y escribir sentencias de SQL. Tampoco es necesario escapar datos o realizar casting. Tan solo es necesario definir la BD en formato XML u obtener la definición desde una BD existente. Se implementa para todos los conceptos clave de la madurez de las capas ORM: el patrón ActiveRecord, validadores, los comportamientos, la herencia de tabla, una ingeniería inversa de BD existentes, conjuntos anidados, las transacciones anidadas, carga lenta, línea de negocio. Es construido para desarrolladores que necesitan mantener el control de su código:

- ☑ La extensibilidad está en el centro de diseño de Propel, y todo lo que se necesita para personalizar, Propel permite hacerlo en un instante.

- ☑ Si necesita cambiar el Sistema Administrador de Base de Datos Relacionales (RDBMS) en el curso del proyecto y reconstruir el modelo. Propel da soporte para MySQL, PostgreSQL, SQLite, MS SQL y Oracle.
- ☑ El código generado por Propel está bien documentado, es amigable y fácil de usar.

El proyecto de Propel se inició en el 2005, está ampliamente documentado, respaldado por muchos tutoriales en la web, que también se beneficia de una comunidad entusiasta que proporciona un apoyo rápido para los desarrolladores principiantes y hardcore. Es liberado bajo la licencia MIT y libre de utilizar, incluso en aplicaciones comerciales. (14)

Lenguaje DQL, XML, XSL, YML, PHP

DQL (Doctrine Query Language): Es un lenguaje de consulta de objetos creado para ayudar a los usuarios en la recuperación de objetos complejos. Entre sus beneficios se encuentra:

- ☑ Fue diseñado desde sus inicios para recuperar los registros (los objetos) no un conjunto de resultado en las filas.
- ☑ Es portátil en diferentes BD.
- ☑ Tiene algoritmos muy complejos e integrados, como (el algoritmo de límite de registros) que puede ayudar al desarrollador a recuperar objetos de manera eficiente.
- ☑ Es compatible con algunas funciones que pueden ahorrar tiempo si se trata de uno-a-muchos, muchos-a-muchos de datos relacionados con condiciones de búsqueda. (15)

XML (Extensible Markup Language): Especificación/lenguaje de programación desarrollada por World Wide Web Consortium (W3C). Es una versión de SGML, diseñado especialmente para los documentos de la web. Permite que los diseñadores creen sus propias etiquetas, característica que posibilita la definición, transmisión, validación e interpretación de datos entre aplicaciones y organizaciones.
Se puede usar para infinidad de trabajos y aporta muchas ventajas en amplios escenarios. A continuación se presentan algunos de sus usos prácticos:

- ☑ **Comunicación de datos:** Si la información se transfiere en XML, cualquier aplicación podría escribir un documento de texto plano con los datos que estaba manejando en formato XML y otra aplicación recibir esta información y trabajar con ella.

- ☑ **Migración de datos:** Si es necesario mover los datos de una BD a otra sería muy sencillo si las dos trabajan en formato XML.
- ☑ **Aplicaciones web:** Hasta ahora cada navegador interpreta la información a su manera y los programadores web tienen que determinar los procesos a utilizar en función del tipo de navegador que utilice el usuario. Con XML es una sola aplicación que maneja los datos y para cada navegador o soporte se podrá tener una hoja de estilo o similar para aplicarle el estilo adecuado. Si mañana la aplicación debe correr en WAP solo se genera una nueva hoja de estilo o símil. (16)

XML, con todas las tecnologías relacionadas, representa una manera distinta de hacer las cosas, más avanzada, cuya principal novedad consiste en permitir compartir los datos con los que se trabaja a todos los niveles, por todas las aplicaciones y soportes. Juega un papel importantísimo en este mundo actual, que tiende a la globalización y la compatibilidad entre los sistemas, ya que es la tecnología que permitirá compartir la información de una manera segura, fiable y fácil. Además, XML permite al programador y a los soportes dedicar sus esfuerzos a las tareas importantes cuando trabaja con los datos, ya que algunas tareas tediosas como la validación de estos o el recorrido de las estructuras corren a cargo del lenguaje y está especificado por el estándar, de modo que el programador no tiene que preocuparse por ello.

Los programas que utilizan el formato XML pueden intercambiar fácilmente sus datos, ya que responden a una misma lógica interna.XML son ficheros de texto que contienen la información organizada en forma de árbol: cada rama puede tener varios atributos propios y servir de base para otras ramas. Además, los documentos XML se pueden transformar (por ejemplo, a formato HTML, para mostrar la información en una página web), o combinar: un tronco con todas sus ramas puede pasar a ser una rama de otro árbol mayor.

XSL (lenguaje de hojas extensibles): Expresión inglesa traducible como "lenguaje extensible de hojas de estilo" es una familia de lenguajes basados en el estándar XML que permite describir cómo la información contenida en un documento XML cualquiera debe ser transformada o formateada para la presentación en un medio.

Esta familia está formada por tres lenguajes:

- ☑ **XSLT (lenguaje de hojas extensibles de transformaciones):** Lenguaje de hojas extensibles de transformación, que permite convertir documentos XML de una sintaxis a otra (por ejemplo, de un XML a otro o a un documento HTML).
- ☑ **XSL-FO (lenguaje de hojas extensibles de formateo de objetos):** Permite especificar el formato visual con el cual se quiere presentar un documento XML, es usado principalmente para generar documentos PDF.
- ☑ **XPath, o XML Path Language:** Es una sintaxis (no basada en XML) para acceder o referirse a porciones de un documento XML.

XSLT o Transformaciones XSL es un estándar de la organización W3C que presenta una forma de transformar documentos XML en otros e incluso a formatos que no son XML. Las hojas de estilo XSLT aunque el término de hojas de estilo no se aplica sobre la función directa del XSLT realizan la transformación del documento mediante una o varias reglas de plantilla. Estas reglas de plantilla unidas al documento fuente a transformar alimentan un procesador de XSLT, que realiza las transformaciones deseadas poniendo el resultado en un archivo de salida, o, como en el caso de una página web, las hace directamente en un dispositivo de presentación tal como el monitor del usuario.

Actualmente, XSLT es muy usado en la edición web y en el desarrollo páginas HTML o XHTML. La unión de XML y XSLT permite separar contenido y presentación, característica que aumenta así la productividad. (17)

YAML: Es un **lenguaje de marcado** "ligero" que permite especificar estructuras (tales como arrays) con menos caracteres que XML, de forma sencilla. Es usado en framework de PHP, dada su facilidad de uso utiliza una notación basada en la indentación y/o un conjunto de caracteres sigil distintos de los que se usan en XML, ventaja que hace fácil componer ambos lenguajes.

Los contenidos en YAML se describen mediante el conjunto de caracteres imprimibles de Unicode. La estructura del documento se denota indentando con espacios en blanco; sin embargo no se permite el uso de caracteres de tabulación para indentar. Los miembros de las listas se denotan encabezados por un guión (-) con un miembro por cada línea, o bien entre corchetes ([]) y separados por coma espacio (,**)**.

Los arrays asociativos se representan mediante los dos puntos seguidos por un espacio. En la forma "clave: valor", bien uno por línea o entre llaves ({}) y separados por coma seguida de espacio (,).

Un valor de un array asociativo viene precedida por un signo de interrogación (?), lo que permite que se construyan claves complejas sin ambigüedad. Los valores sencillos (o escalares) por lo general aparecen sin entrecomillar, pero pueden incluirse entre comillas dobles (**"**), o comillas simples (**'**).En las comillas dobles, los caracteres espaciales se pueden representar con secuencias de escape similares a las del lenguaje de programación C, que comienzan con una barra invertida (****).Se pueden incluir múltiples documentos dentro de un único flujo, separándolos por tres guiones (**---**); los tres puntos (**...**) indican el fin de un documento dentro de un flujo. Los nodos repetidos se pueden denotar con un ampersand (&) y ser referidos posteriormente mediante el asterisco (*). Los comentarios vienen encabezados por la almohadilla (#) y continúan hasta el final de la línea. Los nodos pueden etiquetarse con un tipo o etiqueta mediante el signo de exclamación (**!**) seguido de una cadena que puede ser expandida en una URL. (18)

PHP

PHP es el acrónimo de Hipertext Preprocesor. Es un lenguaje de programación del lado del servidor gratuito e independiente de plataforma, rápido, con una gran librería de funciones y mucha documentación. Lenguaje del lado del servidor que se ejecuta en el servidor web, justo antes de que se envíe la página a través de Internet al cliente.

Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red, y otras tareas para crear la página final que verá el cliente. El cliente solamente recibe una página con el código HTML resultante de la ejecución de la PHP. Como la página resultante contiene únicamente código HTML, es compatible con todos los navegadores.

Como producto de código abierto, PHP goza de la ayuda de un gran grupo de programadores, permitiendo que los fallos de funcionamiento se encuentren y se reparan rápidamente. El código se pone al día continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP. Es utilizado en aplicaciones Web-relacionadas por algunas de las organizaciones más prominentes tales como Mitsubishi, Redhat, Der Spiegel, MP3-Lycos, Ericsson y NASA.

PHP es la opción natural para los programadores en máquinas con Linux que ejecutan servidores web con Apache, pero funciona igualmente bien en cualquier otra plataforma de UNIX o de Windows, con el software de Netscape o del web server de Microsoft. PHP también utiliza las sesiones de HTTP, conectividad de Java, expresiones regulares, LDAP, SNMP, IMAP y protocolos de COM (bajo Windows). (19)

En este primer capítulo se profundizó en el estado del arte de las tecnologías relacionadas con el acceso y almacenamiento de información existente en el mundo, sus principales características, así como sus ventajas y desventajas. El estudio de las principales herramientas y metodologías ha proporcionado elementos que justifican la elección de las técnicas empleadas en el desarrollo de la estandarización del modelo de datos de SIGICEM v1.0.

Capítulo 2: Análisis y descripción de la solución propuesta

El presente capítulo describe, la estrategia de integración con el Framework Symfony, la descripción de la arquitectura a utilizar, una breve descripción de las tablas más significativas y los valores que almacenan las mismas.

2.1 Integración con el Framework Symfony

La lógica del negocio de las aplicaciones web depende de su modelo de datos. Las BD son de modelos relacionales, PHP 5 y Symfony están orientados a objetos, por lo que para acceder de forma efectiva a la BD desde un contexto orientado a objetos necesita de un componente. El componente que se encarga por defecto de gestionar el modelo de datos es realizado mediante el ORM Propel o Doctrine. Las aplicaciones web que son desarrolladas mediante el Framework Symfony, el acceso y la modificación de los datos almacenados en la BD se realizan mediante objetos, de esta forma nunca se accede de forma explícita a ella. Este comportamiento permite un alto nivel de abstracción y una fácil portabilidad del modelo de datos.

El ORM posibilita la reutilización del código, característica que le permite llamar a los métodos de un objeto de datos desde varias partes de la aplicación e incluso desde diferentes aplicaciones, estas características son proporcionadas al realizar la sustitución de registros por objetos y las tablas por clases, esto permite añadir métodos accesores en los objetos que no tienen relación directa con una tabla. Si se dispone por ejemplo de una tabla llamada cliente con dos campos llamados nombre y apellidos, puede que se necesite un dato llamado NombreCompleto que incluya y combine el nombre y los apellidos. En el mundo orientado a objetos, es tan fácil como añadir un método accesor a la clase Cliente. Desde el punto de vista de la aplicación, no existen diferencias entre los atributos Nombre, apellidos y NombreCompleto de la clase Cliente. Solo la propia clase es capaz de determinar si un atributo determinado se corresponde con una columna de la BD.

Al crear las BD, se utilizan diferentes variantes del lenguaje SQL. Si se cambia a otro SGBD, es necesario reescribir parte de las consultas SQL que se definieron para el sistema anterior. Si se crean las consultas mediante una sintaxis independiente de la BD y un componente (PDO) externo se encarga de traducirlas al lenguaje SQL concreto de la misma, se puede cambiar fácilmente de un gestor a otro. Este es precisamente el objetivo de las capas de abstracción. Esta capa obliga a utilizar una sintaxis específica

para las consultas y a cambio realiza el trabajo sucio de optimizar y adaptar el lenguaje SQL a la BD concreta que se está utilizando.

La ventaja más atractiva de la capa de abstracción es la portabilidad, porque hace posible el cambiar la aplicación de un gestor a otro, incluso a la mitad de un proyecto en desarrollo. Si se debe desarrollar rápidamente un prototipo de una aplicación y el cliente no ha decidido todavía la BD que mejor se ajusta a sus necesidades, se puede construir la aplicación mediante SQLite y si el cliente ya se decidió, cambiar fácilmente a MySQL, PostgreSQL u Oracle. Solamente es necesario cambiar una línea en un archivo de configuración y todo funciona correctamente.

Symfony utiliza Propel como ORM y Propel utiliza PDO (PHP Data Objects) como capa de abstracción de BD. Estos 2 componentes externos han sido desarrollados por el equipo de Propel, y están completamente integrados con el Framework Symfony, por lo que se pueden considerar una parte más del framework.

Para generar el modelo de objetos de datos que utiliza el Framework Symfony, se debe traducir el modelo relacional de la BD a un modelo de objetos de datos. Para realizar ese mapeo o traducción, el ORM necesita una descripción del modelo relacional, que se llama "esquema" (schema). En el esquema se definen las tablas, sus relaciones y las características de sus columnas. La sintaxis que Symfony utiliza para definir los esquemas hace uso del formato YAML. Los archivos schema.yml deben guardarse en el directorio mi proyecto/config/. (20)

En el archivo schema.yml, la primera clave representa el nombre de la conexión. Puede contener varias tablas, cada una con varias columnas. A través de la sintaxis de YAML, las claves terminan con dos puntos (:) y la estructura se define mediante la indentación con espacios, no con tabuladores. Cada tabla puede definir varios atributos, incluyendo el atributo phpName (que es el nombre de la clase PHP que será generada para esa tabla). Si no se menciona el atributo phpName para una tabla, Symfony crea una clase con el mismo nombre que la tabla al que se aplica las normas del CamelCase.

Las columnas también pueden definir el atributo phpName, que es la versión modificada de su nombre según las convenciones habituales (Id, Título, Contenido) y que normalmente no es necesario redefinir. Las tablas también pueden definir claves externas e índices de forma explícita, además de incluir definiciones específicas de su estructura para ciertas BD.

El esquema se utiliza para construir las clases del modelo que necesita la capa del ORM. Para reducir el tiempo de ejecución de la aplicación, estas clases se generan mediante una tarea de línea de instrucción:

Php Symfony propel: build-model

Después de construir el modelo, es necesario borrar la caché interna de Symfony mediante la instrucción php Symfony cc para que este sea capaz de encontrar los nuevos modelos. Al ejecutar esa instrucción, se analiza el esquema y se generan las clases base del modelo, que se almacenan en el directorio lib/model/om/ del proyecto:

- ☑ BaseArticulo.php
- ☑ BaseArticuloPeer.php
- ☑ BaseComentario.php
- ☑ BaseComentarioPeer.php

Además, se crean las verdaderas clases del modelo de datos en el directorio lib/model/:

- ☑ Articulo.php
- ☑ ArticuloPeer.php
- ☑ Comentario.php
- ☑ ComentarioPeer.php

Aunque el modelo de datos es independiente de la BD utilizada, es necesario utilizar una BD concreta. La información mínima que necesita Symfony para realizar peticiones a la BD es su nombre, las credenciales para acceder (usuario, contraseña) y el tipo que va a utilizar. (21)

2.2 Descripción de la arquitectura

El SIGICEM es un proyecto complejo debido a la cantidad de funcionalidades que presenta, entre las cuales se encuentra el control de todos los equipos médicos del país. El mismo es de vital importancia porque mediante este se elaboran estadísticas de las piezas que hay que recopilar para la renovación de equipos médicos que están en mal estado y que pueden seguir brindando servicios a la población, debido a esto es necesario para la implementación de este sistema utilizar herramientas que faciliten un desarrollo rápido y seguro. Los desarrolladores de SIGICEM utilizan como lenguaje de programación PHP

5.2.3 y Symfony como framework de desarrollo, el cual utiliza el estilo arquitectónico MVC (Modelo-Vista-Controlador):

- ☑ **Modelo:** Esta es la representación específica de la información con la cual el sistema opera. En resumen, el modelo se limita a lo relativo de la vista y su controlador facilitando las presentaciones visuales complejas. El sistema también puede operar con más datos no relativos a la presentación, haciendo uso integrado de otras lógicas de negocio y de datos afines con el sistema modelado.
- ☑ **Vista:** Este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.
- ☑ **Controlador:** Este responde a eventos, usualmente acciones del usuario, e invoca peticiones al modelo y, probablemente, a la vista.

Este MVC es la base del funcionamiento del Framework Symfony, es fácilmente integrable a las aplicaciones debido a su composición y que contiene diferentes clases de gran utilidad. Este framework también presenta características en las que se encuentra:

- ☑ Fácil de instalar y configurar en la mayoría de plataformas.
- ☑ Independiente del sistema gestor de bases de datos. Su capa de abstracción y el uso de Propel, permiten cambiar con facilidad de SGBD en cualquier fase del proyecto.
- ☑ Utiliza programación orientada a objetos, de ahí que sea imprescindible PHP 5.
- ☑ Sencillo de usar en la mayoría de casos, aunque es preferible para el desarrollo de grandes aplicaciones Web que para pequeños proyectos.
- ☑ Aunque utiliza MVC (Modelo Vista Controlador), tiene su propia forma de trabajo en este punto, con variantes del MVC clásico como la capa de abstracción de base de datos, el controlador frontal y las acciones.
- ☑ Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- ☑ Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- ☑ Código fácil de leer que incluye
- ☑ Fácil de extender, lo que permite su integración con las bibliotecas de otros fabricantes.

- ☑ Una potente línea de comandos que facilitan generación de código, lo cual contribuye a ahorrar tiempo de trabajo

La parte del Modelo se puede hacer con la ayuda del ORM Propel. Este basándose en la descripción de la BD, genera:

- ☑ **Clases**: característica que le permite a Symfony lograr su programación orientada a objeto,
- ☑ **Formularios:** Clases echas de campos que brindan al programador las herramientas necesarias para mostrar y validar los formularios que están íntimamente vinculado con la BD.
- ☑ **Filtros.:** Mecanismos de seguridad, ya que por el debe pasar cada petición antes de ser ejecutada la opción
- ☑ **Sentencias SQL:** Sentencias SQL para la interacción con el modelo de datos.

La configuración de la BD se puede hacer con una tarea o editando el archivo database.yml de configuración. Además de su configuración, también es posible hacer la inyección inicial de datos, gracias a los archivos de datos.

El modelo es responsable de:

- ☑ Acceder a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento.
- ☑ Define las reglas de negocio (la funcionalidad del sistema). Lleva un registro de las vistas y controladores del sistema.
- ☑ Ante un modelo activo, notificará a las vistas los cambios que en los datos pueda producir un agente externo.

Propel que es un potente y completo ORM para PHP 5.2.3 o posterior. La principal ventaja que aporta el ORM es la reutilización, permitiendo llamar a los métodos de un objeto de datos desde varias partes de la aplicación e incluso desde diferentes aplicaciones. La capa ORM también encapsula la lógica de los datos. La utilización de objetos en vez de registros y de clases en vez de tablas, tiene otra ventaja:

- ☑ permite añadir métodos asesores en los objetos que no tienen relación directa con una tabla.

Su principal ventaja radica en poder acceder a la BD mediante la programación orientada a objetos, tiene su propio lenguaje de consultas y trabaja de manera rápida y eficiente. Permite exportar una BD existente

a sus clases correspondientes y viceversa, es decir, convertir clases (convenientemente creadas por las pautas del ORM) a tablas de una BD. Este corresponde a los tipos de BD: MySQL, MS SQL, PgSQL, SQLite y Oracle.

2.3 Requisitos No Funcionales

Usabilidad.

- ☑ Preparar a los Administradores en la estandarización de modelos de datos.
- ☑ Acceso fácil y rápido a los distintos tipos de gestores de BD.

Rendimiento.

- ☑ La aplicación debe responder en un tiempo relativamente rápido a las peticiones del usuario (menos de 3 segundos).
- ☑ La aplicación necesita un servidor de BD en una PC o servidor con (1giga o superior) de RAM y 1 disco duro de 250 Gigas.

Requerimientos de Hardware.

PC Servidor Web: Software instalado en el Servidor Web.

- ☑ Servidor Web Apache 2.2 o superior

Servidor de BD: Software instalado en el Servidor de BD.

- ☑ Servidor de BD que se va a utilizar.
- ☑ Servidor de BD (Intel Pentium IV o superior. D 2x2 cache. 3.00 GHz, Memoria RAM 1 GB, Almacenamiento en discos de 250 GB)

Requisitos de Seguridad

Seguridad del sistema

- ☑ **Confidencialidad:** La información manejada por el sistema está protegida de acceso no autorizado y divulgación.
- ☑ **Integridad:** La información manejada por el sistema es objeto de cuidadosa protección contra la corrupción y estados inconsistentes. Se incluyen también mecanismos de chequeo de integridad y realización de auditorías por personal calificado de la entidad.

- ☑ **Disponibilidad:** Los usuarios autorizados (autenticados por dominio y según su rol) se les garantizará el acceso a la información, los dispositivos o mecanismos utilizados para lograr la seguridad no ocultarán o retrasarán a los usuarios para obtener los datos deseados en un momento dado.

2.4 Modelo de datos

El modelo de datos de SIGICEM, actualmente está formado por 112 tablas que brindan soporte para el almacenamiento de toda la información que este sistema maneja. Estas pueden ser agrupadas, según la información que gestionan, pudiéndose mencionar: ubicación, nomencladores, grupos técnicos, equipo, despiece, reporte de pieza, despacho, orden de despacho, especialista, solicitud baja, recepción, oferta y solicitud de compra. Esto le proporciona a los implementadores del sistema una organización de los datos para un desarrollo eficiente del sistema.

A continuación se ofrece la descripción de las tablas más significativas. En cada una de ellas se especifican varias de las propiedades básicas que estas poseen, como son el nombre, atributos, tipo de datos, y descripción tanto de la clase como de los atributos.

Nombre:	Unidad de Salud	
Descripción:	Registra información de la unidad de salud	
Atributos	**Tipo**	**Descripción**
id_unidad_salud	VARCHAR	Identificador de la unidad de salud.
id_tipo_unidad_salud	INTEGER	Identificador del tipo de unidad de salud.
id_tipo_institucion	INTEGER	Identificador del tipo de institución.
id_nivel_atencion	INTEGER	Identificador del nivel de atención.
id_localidad	INTEGER	Identificador de la localidad.
id_area_salud	VARCHAR	Identificador del área de salud.
id_subordinacion	INTEGER	Identificador de subordinación.
id_organismo_pertenece	INTEGER	Identificador del organismo al que pertenece.
id_grupo_tecnico	INTEGER	Identificador del grupo técnico.
nombre_unidad	VARCHAR	Nombre de la unidad de salud.
Dirección	TEXT	Datos de la dirección.

Tabla # 1: Descripción de la tabla Unidad de Salud.

Nombre:	Pieza
Descripción:	Registra datos de las piezas.

Atributos	Tipo	Descripción
id_pieza	INTEGER	Identificador de la pieza.
id_nomenclador_equipo	INTEGER	Identificador del nomenclador equipo.
id_unidad_medida	INTEGER	Identificador de la unidad de medida.
descripcion	VARCHAR	Descripción de la pieza.
ref_fab	VARCHAR	Referencia de la fábrica de la pieza.
gastable	VARCHAR	Si es o no gastable.
estandar	VARCHAR	Si es estándar para varios equipos.
precio_unitario	VARCHAR	Precio unitario de la pieza.
Código	VARCHAR	Código de la pieza.
id_pieza	INTEGER	Identificador de la pieza.

Tabla # 2: Descripción de la tabla Pieza.

Nombre:	Especialista.	
Descripción:	Registra datos de los especialistas	
Atributos	**Tipo**	**Descripción**
id_especialista	VARCHAR	Identificador del especialista.
id_grupo_tecnico_brigada	INTEGER	Identificador del grupo técnico de la brigada.
id_especialidad_atiende	INTEGER	Identificador del especialista que atiende.
id_municipio	INTEGER	Identificador del municipio.
id_cargo	INTEGER	Identificador del cargo.
Nombre	VARCHAR	Nombre del especialista.
apellido1	VARCHAR	Apellido del especialista
apellido2	VARCHAR	Apellido del especialista.
Ci	VARCHAR	Carnet de identidad del especialista.
id_especialista	VARCHAR	Identificador del especialista.

Tabla # 3: Descripción de la tabla Especialista.

Nombre:	Equipo.	
Descripción:	Registra Información del equipo	
Atributos	**Tipo**	**Descripción**
id_equipo	VARCHAR	Identificador del equipo.
id_nomenclador_equipo	INTEGER	Identificador del nomenclador equipo.
id_unidad_salud	VARCHAR	Identificador de la unidad de salud.

id_departamento	INTEGER	Identificador del departamento.
id_servicios	INTEGER	Identificador de los servicios.
id_especialista	VARCHAR	Identificador del especialista.
id_estado	INTEGER	Identificador del estado.
id_pais_adquisicion	BIGINT	Identificador del país de adquisición.
anno_fabricante	VARCHAR	Fecha y el fabricante del equipo.
fecha_instalacion	DATE	Fecha de instalación.
Noserie	VARCHAR	Número de serie del equipo.
Noinventario	VARCHAR	Número de inventario del equipo.
detalle_ubicacion	TEXT	Detalles de la ubicación del equipo.
observaciones	TEXT	Observaciones hechas al equipo.
codigo	VARCHAR	Código del equipo.
es_sistema	VARCHAR	Si es o no un sistema.
precio_unitario	VARCHAR	Precio unitario del equipo.
fecha_adquisicion	DATE	Fecha de adquisición del equipo.

Tabla # 4: Descripción de la tabla Equipo.

Nombre:	Nomenclador equipo.	
Descripción:	Registra datos de los nomencladores de equipo	
Atributos	**Tipo**	**Descripción**
id_nomenclador_equipo	INTEGER	Identificador del nomenclador de equipo.
id_especialidad	INTEGER	Identificador del especialista.
id_tipo_equipo	INTEGER	Identificador del tipo de equipo.
id_denominacion	INTEGER	Identificador de la denominación.
id_fabricante	INTEGER	Identificador del fabricante.
id_marca	INTEGER	Identificador de la marca.
id_modelo	INTEGER	Identificador del modelo.
codigo	VARCHAR	Identificador del código.

Tabla # 5: Descripción de la tabla Nomenclador Equipo.

Nombre:	Orden de Servicio.	
Descripción:	Registra información de la orden de servicio.	
Atributos	**Tipo**	**Descripción**

id_orden_servicio	INTEGER	Identificador de la orden de servicio
id_orden_servicio_estado_final	INTEGER	Identificador de la orden de servicio en su estado final.
id_estado	INTEGER	Identificador del estado.
id_reporte_equipo	VARCHAR	Identificador del reporte del equipo.
id_equipo	VARCHAR	Identificador del equipo.
id_tipo_trabajo	INTEGER	Identificador del tipo de trabajo.
fecha_inicio	DATE	Fecha de inicio de la orden.
observaciones	TEXT	Observaciones de la orden.
fecha_terminado	DATE	Fecha de terminación de la orden.
fecha_entrega	DATE	Fecha de entrega de la orden.
nombre_usuario_conforme	VARCHAR	Nombre del usuario conforme.
ci_usuario_conforme	VARCHAR	Carnet de identidad del usuario conforme.
codigo_os	VARCHAR	Código de la orden de servicio.
causas	TEXT	Causas de la orden de servicio.
desperfecto_reportado	TEXT	Desperfectos reportados.

Tabla # 6: Descripción de la tabla Orden de servicio.

Nombre:	mantenimiento	
Descripción:	Registra información del mantenimiento	
Atributos	**Tipo**	**Descripción**
id_mantenimiento_procedimiento	INTEGER	Identificador del procedimiento de mantenimiento
id_mantenimiento_estado_ejecucion	INTEGER	Identificador del estado de ejecución
id_mantenimiento_frecuencia	INTEGER	Identificador de la frecuencia del mantenimiento
id_mantenimiento_tipo_planificacion	VARCHAR	Identificador del tipo de planificación.
id_equipo	VARCHAR	Identificador del equipo.
id_meses	INTEGER	Identificador de los meses.
id_especialista	VARCHAR	Identificador del especialista
observaciones	TEXT	Observaciones de la orden.
fecha_planificacion	DATE	Fecha de planificación
fecha_inicio	DATE	Fecha de inicio
fecha_fin	DATE	Fecha de fin

Tabla # 7: Descripción de la tabla Mantenimiento.

Nombre:	Reporte Equipo.	
Descripción:	Registra Información del equipo en reporte	
Atributos	**Tipo**	**Descripción**
id_equipo	VARCHAR	Identificador del equipo.
id_estado_interes	INTEGER	Identificador del estado.
id_estado_reportado	INTEGER	Identificador del estado reportado.
id_tipo_entrada_registro	INTEGER	Identificador del tipo de entrada del registro.
persona_reporta	VARCHAR	Persona que reporta.
fecha_reporte	DATE	Fecha del reporte
causas	TEXT	Motivos, causas del reporte
desperfecto_reportado	TEXT	Desperfecto reportado.
codigo_reporte	VARCHAR	Código del reporte.
recepcionista_recibe_reporte	VARCHAR	Recepcionista que recibe reporte.
nombre_recoge	VARCHAR	Nombre del que recoge el reporte.
ci_recoge	VARCHAR	Carnet de identidad.
fecha_entregado	DATETIME	Detalles de la ubicación del equipo.
nombre_recepcionista_entrega	VARCHAR	Nombre de la recepcionista.
observaciones	TEXT	Observaciones.
id_estado_interes	INTEGER	Estado de interés.
id_estado_reportado	INTEGER	Estado reportado.

Tabla # 8: Descripción de la tabla Reporte Equipo.

Nombre:	Solicitud de Compra	
Descripción:	Registra Información de las solicitudes de compra.	
Atributos	**Tipo**	**Descripción**
id_oferta	INTEGER	Identificador de la oferta
id_solicitud_compra_estado	INTEGER	Identificador de la solicitud de compra.
id_especialista	VARCHAR	Identificador del especialista.
numero_registro	INTEGER	Número de registro.
consumidor	VARCHAR	Consumidor.
fecha_confecion	DATE	Fecha de confección.
fecha_recepcion	DATE	Fecha de recepción
aprobado_por	VARCHAR	Por quien es aprobado.
recibido_por	VARCHAR	Quien lo recibe.

cantidad	INTEGER	Cantidad.
suministrador	VARCHAR	Suministrador.
fecha_pago	DATE	Fecha de pago.
fecha_finanza	DATE	Fecha de Finanza.
id_oferta	INTEGER	Identificador de la oferta.
id_especialista	VARCHAR	Identificador de especialista.

Tabla # 9: Descripción de la tabla Solicitud de compra.

Nombre:	Oferta Pieza	
Descripción:	Registra Información de las ofertas de las piezas.	
Atributos	**Tipo**	**Descripción**
id_oferta	INTEGER	Identificador de la oferta
id_oferta_descripcion	INTEGER	Identificador de la oferta con su descripción.
id_oferta_modelo	INTEGER	Identificador de la oferta y el modelo.
codigo	VARCHAR	Número de registro.
referencia	VARCHAR	Referencia.
cantidad	INTEGER	Cantidad.
precio	VARCHAR	Precio.
id_oferta_descripcion	INTEGER	Identificador de la descripción.

Tabla # 10: Descripción de la tabla Solicitud de compra.

En el capítulo 2 se ha descrito parte del modelo de datos propuesto por el SIGICEM, una breve descripción de la arquitectura así como la integración del modelo de datos con el Framework Symfony. Se detallan además las entidades más significativas de la BD.

Capítulo 3: Implementación y pruebas

En este capítulo se ofrece información sobre la validación teórica y funcional del modelo de datos propuesto por el SIGICEM. Se detallan algunos elementos necesarios para la creación de un plugin para Symfony y se analiza la solución propuesta así como las pruebas realizadas a la misma.

3.1 Validación teórica del diseño

La validación teórica del diseño incluye fundamentalmente un análisis muy detallado de la integridad de la información, característica altamente deseada, porque asegura la calidad de almacenamiento y disponibilidad de los datos, con su tratamiento se evita errores de entrada introducidos por los usuarios descuidados o cualquier otra circunstancia de intento de violar la información existente en la BD.

Integridad

La integridad es uno de los factores más importantes a la hora de realizar el diseño de una BD. Esta se divide en varios aspectos como son:

- ☑ **Integridad referencial:** Garantiza interrelaciones válidas entre entidades. Implica que los datos sean correctos, sin duplicaciones, pérdida de datos o relaciones mal resueltas. Todas las bases de datos relacionales incluyen ésta propiedad pues el software gestor es responsable de su cumplimiento.
- ☑ **Integridad de Dominio:** La integridad de dominio viene dada por la validez de las entradas de los datos para una columna determinada. Se puede exigir la integridad de dominio para restringir el tipo mediante tipos de datos, el formato mediante reglas y restricciones CHECK, o el intervalo de valores posibles mediante restricciones FOREIGN KEY, definiciones DEFAULT, definiciones NOT NULL.

- ☑ **Integridad de la Entidad:** define una fila como entidad única para una tabla determinada. La integridad de entidad exige la integridad de las columnas de los identificadores o la clave principal de una tabla, mediante índices y restricciones UNIQUE, o restricciones PRIMARY KEY.[19]

Restricción de PRIMARY KEY (llave primaria)

Toda entidad debe tener una llave primaria la cual identifica una tupla unívocamente de las demás.

3.2 Descripción de la estructura de un plugin

En ocasiones, es necesario reutilizar una porción de código desarrollado para alguna aplicación realizada mediante el Framework Symfony. Si se puede encapsular ese código en una clase, tan sólo es necesario guardar la clase en algún directorio lib/ para que otras aplicaciones puedan encontrarla, esto constituye un tipo de aplicación también conocido como plugin.

Un Plugin es una extensión encapsulada para un proyecto desarrollado mediante el Framework Symfony. Estos posibilitan no solamente reutilizar código propio, sino que permiten además aprovechar los desarrollos realizados por otros programadores y añadir al núcleo de Symfony extensiones realizadas por otros desarrolladores. (22)

Los plugin permiten encapsular clases, filtros, mixins, helpers, archivos de configuración, tareas, módulos, esquemas y extensiones para el modelo, fixtures y archivos estáticos. Estas son fáciles de instalar, actualizar y de desinstalar. Se pueden distribuir en forma de archivo comprimido en tgz, en paquete pear o directamente desde el repositorio de código. La forma en la que Symfony carga los plugins permite que los proyectos puedan utilizarlos como si fueran parte del propio framework.

3.3 Creación de un plugin

Los plugins se crean mediante el lenguaje PHP. Si se entiende la estructura de una aplicación, es posible comprender la estructura de un plugin. Su directorio se organiza de forma muy similar al directorio de un proyecto (ANEXO 1). Si se ha creado una nueva característica para Symfony, puede ser útil encapsularla en un componente para poder utilizarla en otros proyectos. El primer paso es el de organizar los archivos de forma lógica para que los mecanismos de carga automática de Symfony puedan cargarlos en el momento necesario.

Los plugins o aplicaciones permiten encapsular clases, filtros, archivos de configuración, tareas, módulos, esquemas, extensiones para el modelo, fixtures, archivos estáticos, tasks entre otros. Este es sólo una estructura de directorios con los archivos organizados en una estructura previamente definida, según la naturaleza de los archivos que se necesite para ejecutar cierta función dentro de un proyecto de Symfony. En este caso, se creará un plugin que contenga los archivos de configuración y la hoja de estilos XSL, el plugin se llamará xmlaymlPlugin.

Lo que se necesita para crear esta aplicación, es un directorio que se llame xmlaymlPlugin y crear allí la estructura de directorios que se necesite, es decir, la carpeta bin con el archivo db4.sh, la carpeta lib\task donde se incluirá el archivo php llamado dxmlaymlTask.class el cual lleva el control de las transformaciones que se le aplican al documento XML desarrollado mediante la herramienta DBDesigner Fork , el control de estas transformaciones se realizan mediante funciones implementadas en el lenguaje php y es el encargado de devolver el archivo yml estandarizado para Propel.

Además en la carpeta lib\vendor\db2 se encontrará la hoja de estilos XSL xslaxml.xsl. Esta consigue separar la información (almacenada en un documento XML) de su presentación, usando en cada caso las transformaciones que sean necesarias para que el contenido aparezca de la forma más adecuada. Mediante la hoja de estilo XSLT se pueden usar diferentes hojas de estilo, o incluso la misma, para presentar la información de diferentes maneras dependiendo de los deseos o de las condiciones del usuario.

Las hojas de estilos xsl son capaces de interactuar con conjuntos de nodos devueltos por determinada expresión XPath y:

- ☑ Ordenarlos.
- ☑ Implementar procesamiento condicional.
- ☑ Copiar nodos.
- ☑ Declarar variables.

Características de los elementos XSLT

- ☑ **xsl: template;** A través de su propiedad match se establece el criterio para la selección de nodos a procesar.
- ☑ **xsl: element:** Permite la creación de un nuevo elemento cuyo identificador se especifica en la propiedad name.
- ☑ **Toma valores a través de expresiones XPath:** estas se encierran en ({}).

Realizan la transformación del documento utilizando una o varias reglas de plantilla. Estas reglas de plantilla unidas al documento fuente transforman y alimentan un procesador de XSLT, y realiza las transformaciones deseadas poniendo en resultado un archivo de salida.

XSLT o XSL Transformaciones es un estándar que presenta una forma de transformar documentos XML en otros, incluso a formatos que no son XML. Realizan la transformación del documento utilizando una o varias reglas de plantilla unidas al documento fuente a transformar. Actualmente, XSLT es muy usado en la edición web, generando páginas HTML o XHTML. La unión de XML y XSLT permite separar contenido y presentación, aumentando así la productividad. A continuación se presentan algunas características de este:

XSLT Características:

- ☑ XSL: lenguaje de hojas extensibles.
- ☑ XSL-FO es la especificación que desarrolla el lenguaje de formateo.
- ☑ XSL consiste en tres partes:
    - ☑ XSLT – lenguaje para la transformación de documentos XML.
    - ☑ XPath – lenguaje para la navegación en documentos XML.
    - ☑ XSL-FO – lenguaje para el formateo de documentos XML.
- ☑ XSLT es la especificación que desarrolla el lenguaje de transformación. Hace uso de la especificación XPath. Ha sido diseñada para ser utilizada de forma independiente aunque es utilizada desde la especificación XSL.
- ☑ XSLT es un lenguaje de programación de hoja de estilos para la transformación de documentos XML en otros documentos XML, HTML, XHTML, WML o incluso PDF.
- ☑ XPath es la especificación que desarrolla el lenguaje para acceder a los elementos de un documento XML. Ha sido desarrollada para ser utilizada desde la especificación XSLT y XPointer.

- ☑ XSLT, XSL, XSL-FO y XPath están definidos en XML
- ☑ XSLT, XSL, XPath y XSL-FO son recomendaciones del W3C.

A través de estas características XSLT permite:

- ☑ Formatear los elementos fuente basados en relaciones de ancestro/descendiente, posición y unicidad.
- ☑ La creación de construcciones de formato sofisticadas incluyendo texto generado e imágenes
- ☑ La definición de macros de formateo reutilizables
- ☑ Estilos independices de la dirección en que se escriba el lenguaje
- ☑ Conjunto de objetos de formato extensible
- ☑ Una manera de describir el proceso de transformación es decir que las XSLT transforma un árbol XML de entrada e otro árbol XML de salida.

Esto se logra debido que la programación basada en XSLT se fundamenta mediante arboles lo que permite que:

- ☑ El documento original se traduce a un árbol que representa el documento original:
    - ☑ raíz del árbol -> elemento raíz
    - ☑ hijos de un nodo -> elementos contenidos en él
- ☑ El estilo convierte este árbol en un árbol de objetos de flujo.
- ☑ La conversión se hace a través de reglas de construcción
- ☑ Una regla de construcción de se compone de:
    - ☑ patrón, que especifica a que elementos de árbol fuente se aplica
    - ☑ acción, que indica como se traduce un subárbol a otro
    - ☑ el estilo se procesa recursivamente

Los elementos antes mencionados permiten que le aplicación cumpla con el objetivo con el cual se creo, con el propósito de obtener un modelo de datos estándar para el Framework Symfony. Para habilitar esta

aplicación en cualquier proyecto, sólo se tiene que ubicar en el directorio plugins, luego en el ProjectConfiguration.class.php habilitar el plugin y con esto ya se puede usar el plugin.

El componente xmlaymlPlugin le añade a Symfony una nueva tarea que le permite convertir un esquema.xml desarrollado mediante el programa DB Designer Fork en un archivo schema.yml optimizado para Propel, con este plugin se puede evitar el trabajo de la construcción de su schema.yml de forma manual. La aplicación está formada por 3 archivos, la unión de estos logra un trabajo curioso y eficiente debido a que una vez generado el schema.yml, este posibilita la obtención de un modelo de datos estándar para el Framework Symfony, lo cual a través del ORM Propel permitirá exportar su modelo de datos hacia los distintos gestores de BD existentes. La aplicación está conformada por la siguiente estructura:

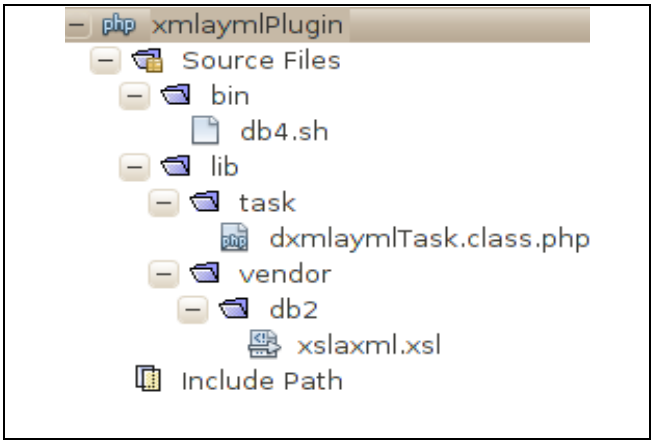


Figura # 1: Estructura de la aplicación.

- ☑ **Archivo dxmlaymlTask.class.php:** Transforma el diseño de la BD desarrollada en el DB Designer Fork (documento.xml) en un esquema.yml.

- ☑ **Archivo xslaxml.xsl:** Carga la hoja de estilo de transformación el cual va transformar un modelo de BD XML, desarrollado mediante el DB Designer Fork en un modelo de datos para Propel.

Para ejecutar la aplicación es necesario tener instalado el Framework Symfony y tener el ORM Propel habilitado debido a que esta aplicación solo funciona con este, se copia la carpeta xmlaymlPlugin dentro de la carpeta plugin del proyecto creado, una vez copiada ahí se específica la dirección donde se encuentra la BD.xml, el nombre que tiene la misma carpeta de salida y el nombre que tendrá la BD,

después se habilita el plugin en la clase ProjectConfiguration.class que se encuentra en la carpeta config de la aplicación del proyecto. Luego de haber realizado estos pasos en la consola mediante la instrucción:

php symfony propel: xml2yml sigicem

Se obtiene un modelo de datos estandarizado para los distintos gestores de BD existentes.

3.4 Validación funcional de la Base de Datos

3.5.1 Ambiente de las pruebas

Las BD donde se realizaron las pruebas llevan como nombre “Prueba”. Se crearon en un servidor con las siguientes características:

- ☑ Sistema Operativo: Ubuntu 10.4
- ☑ Memoria RAM: 512 MB
- ☑ Disco duro: 250
- ☑ Velocidad CPU: 3.0:GHz

Para el llenado de las mismas se utilizó la herramienta Data Generator, herramienta que permite generar datos para una o varias tablas a la vez y respeta la integridad referencial de los datos que provienen de otras tablas para evitar errores. Se crearon volúmenes de 50000 tuplas para las tablas principales (orden de servicio, nomenclador equipo, equipo, especialista, unidad de salud y pieza) donde se realizarían las consultas más específicas. Se elaboraron consultas de tipo SELECT, INSERT y UPDATE de diferente complejidad (ANEXO 2) mediante el lenguaje SQL y se elaboraron de acuerdo con el modelo de datos del SIGICEM. La herramienta utilizada para medir el rendimiento de las consultas fue el JMeter, además se validó el modelo de datos en los SGBD PostgresSQL y MySQL.

3.5.2 Validación del modelo de datos

Una vez ejecutada la aplicación al modelo de datos (sigicem.xml) del SIGICEM, se obtuvo un modelo de datos estándar, se comprobó exportándolo a diferentes SGBD como: Postgres y MySQL, este modelo de datos es compatible al original (al que se diseñó originalmente con la ayuda de DBDesigner Fork), no ocurrió pérdidas de información ni se afectó la integridad de la misma.

Luego de haber aplicado el plugin al esquema.xml del modelo de datos del SIGICEM, se creó una BD en MySQL y en PostgreSQL, luego se les insertaron datos a ambos modelos de datos para realizar pruebas de compatibilidad y rendimiento. Es necesario antes de realizar estas operaciones modificar el dsn, nombre de la BD, usuario y contraseña en el fichero database.yml que Symfony utiliza para establecer la conexión con un SGBD determinado. ANEXO 3

Luego de haberse realizado las configuraciones pertinentes para lograr la comunicación con el gestor MySQL y PostgreSQL respectivamente, se insertaron datos en la BD creada para MySQL y PostgreSQL, mediante la Instrucción:

php symfony propel: insert-sql

Mediante estos SGBD se demostró que las mismas eran compatibles con el modelo de datos diseñado en el DB Designer Fork dejando de esta forma lista la persistencia de la información para que el SIGICEM interactúe a través del ORM Propel con cualquiera de los SGBD mencionados. ANEXO 4

3.5.2.1 Validación del modelo de datos en los gestores de base de datos Mysql y Postgres

La realización del modelo de datos (XML) mediante la herramienta DB Designer Fork, fue ejecutado por la aplicación xmlaymlplugin con el objetivo de obtener un modelo de datos estándar para el Framework Symfony, además, permitir la migración del modelo de datos del SIGICEM hacia los distintos gestores existentes, componente que se creó gracias a la investigación realizada referente a la incógnita de como obtener un modelo de datos estándar, lo que permitió la implementación de dicha aplicación, obteniendo un modelo de datos estándar. Este fue validado en los gestores de base de datos Mysql y Postgres, con el propósito de verificar si este modelo es estándar o no.

En estas validaciones se ejecutó el modelo de datos en los gestores Mysql y Postgres, con el objetivo de insertar los tipos de datos en ambos gestores y verificar el tiempo de repuesta en realizar dichas operaciones. Este modelo fue insertado correctamente en los dos gestores, en un tiempo de 00:06:25 segundos en Mysql y 00:07:35 en Postgres. Las migraciones de modelos de datos anteriormente se realizaban de forma manual, tarea que resultaba muy engorrosa de realizar por los programadores. Mediante la aplicación xmlaymlplugin se reduce el tiempo y agiliza el desarrollo de los sistemas que la utilicen, con el objetivo de obtener un modelo de datos estándar para migrar hacia otros gestores de base de datos, siendo de gran beneficio para los desarrolladores del SIGICEM.

3.5.3 Herramientas para pruebas de carga intensiva (selección de consultas)

Realizado el modelo de datos (sigicem.xml) en el programa DB Designer Fork se le aplicó la aplicación logrando la obtención de un modelo de datos estándar, este se comprobó en los distintos SGBD mysql y Postgres. Una vez realizada esta importante tarea se ejecutaron pruebas de carga sobre el mismo teniendo en cuenta los tiempos de respuestas de los mismos antes de determinadas peticiones definidas por los usuarios del SIGICEM. Estas verifican el tiempo de respuesta del sistema para transacciones, bajo diferentes condiciones de carga. Miden la capacidad del sistema para continuar funcionando apropiadamente bajo diferentes condiciones de carga. La meta de las pruebas de carga es determinar y asegurar que el sistema funciona apropiadamente aún más allá de la carga de trabajo máxima esperada. (23)

Para diseñar pruebas en el JMeter se realiza de la siguiente forma:

El componente principal es denominado Test Plan o Plan de Pruebas donde se definen todos los aspectos relacionados con la prueba. El plan de pruebas muestra automáticamente al abrir la aplicación. Primeramente se crea un ThreadGroup o Grupo de Hilos, considerado como el grupo de usuarios o peticiones que se desea simular para la BD. Cuando es creado el grupo de hilos se llenan las siguientes opciones:

1. **Nombre:** Se define un nombre.
2. **Number of Threads (users):** Equivale al número de usuarios que se desean simular.
3. **Ramp-Up Period:** Es el lapso de tiempo en segundos que se desea tener entre cada grupo de peticiones, se usará en este caso 1.
4. **LoopCount o Forever:** Se utiliza para indicar si la simulación para grupos de hilos, será llevada a cabo infinitamente o por un ciclo determinado de veces.

Una vez definidas las características del Grupo de Hilo se pasa a generar las JDBC Connection Configuration donde se crea una para cada BD a probar. Las JDBC Connection Configuration se usan para configurar las conexiones de las BD. Esta posee los siguientes aspectos:

1. **Nombre de la variable:** El que se desee.
2. **Database URL:** Dirección donde se encuentra la BD en el servidor, ejemplo: jdbc:MySQL://localhost:3306/sigicem

3. **JDBC Driver class:** Para MySQLcom.MySQL.jdbc.Driver **ó** Para PostgreSQLorg.PostgreSQL.Driver
4. **El usuario y contraseña** de la BD.

El resto de los campos que aparecen están predefinidos por defecto. Luego se generan las JDBC Request utilizadas para definir las requisiciones de simulación, en la cual aparecerán las siguientes opciones:

1. **Name:** El que se desee dar a la consulta.
2. **Variable Name:** El mismo que se definió en la variable Name de la JDBC
3. **Tipo de consulta:** SelectStatement en caso de ser una consulta SELECT y así sucesivamente.
4. **Consulta (Query):** Donde se define la consulta.

Luego se generan los Listener para mostrar los resultados. Para la realización de las pruebas se utiliza el listener **SummaryReport,** los datos que presentan son:

- ☑ **Label**: Etiqueta de la muestra
- ☑ **#Muestras:** Cantidad de peticiones realizadas a la BD.
- ☑ **Media:** Tiempo promedio en milisegundos para un conjunto de resultados.
- ☑ **Min:** Tiempo mínimo que demora una petición en acceder a la BD.
- ☑ **Max:** Tiempo máximo que demora una petición en acceder a la BD.
- ☑ **%Error:** Porcentaje de peticiones con errores.
- ☑ **Rendimiento:** Rendimiento medido en las peticiones por segundo / minuto / hora.

Luego se ejecutan las consultas elaboradas para los SGBD que contienen los datos de pruebas con los que va a interactuar la herramienta JMeter, con el objetivo de medir el rendimiento y tiempo de respuestas que se obtiene al realizar cada una de las siguientes consultas al modelo de datos obtenido, comparando los tiempos de respuesta entre los gestores postgres y mysql, para entender en cual es mas eficiente o si son semejantes en sus resultados.

3.7 Resultado de las Pruebas

3.7.1 Consultas de Selección

En el ANEXO 5 se muestra el rendimiento en segundos, así como el número mínimo, máximo y media de las peticiones realizadas para cada una de las 4 consultas de selección. En el ANEXO 6 se muestra un gráfico que representa el rendimiento para cada una de las consultas de selección.

3.7.2 Consultas de Inserción

En el ANEXO 7 y ANEXO 8 se muestra el rendimiento en segundos, así como el número mínimo, máximo y media de las peticiones realizadas para cada una de las 4 consultas de inserción.

3.7.3 Consultas de Actualización

En el ANEXO 9 y ANEXO 10 se muestra el rendimiento en segundos, así como el número mínimo, máximo y media de las peticiones realizadas para cada una de las 4 consultas de actualización.

Todas las consultas se ejecutaron en los SGBD MySQL y Postgres, aunque los resultados no fueron iguales, se pueden apreciar que no son muy diferentes, mostrando estos resultados un comportamiento muy parecido en ambas bases de datos a pesar de estar implementados en diferentes SGBD.

En este capítulo se han realizado las validaciones al diseño propuesto. Se efectuaron pruebas de rendimientos con el objetivo de validar funcionalmente el diseño de la BD del SIGICEM, esta BD se ejecutó en los SGBD MySQL y PostgreSQL, obteniendo resultados positivos teniendo en cuenta la rapidez y rendimiento de las respuestas.

Conclusiones

- ☑ Se realizó un estudio sobre las herramientas y sistemas existentes a nivel internacional y nacional para la estandarización de modelos de datos para establecer las bases de la investigación.

- ☑ Se aplicaron los lenguajes de programación y tecnologías necesarias para la implementación (PHP, XML, XSL, YML, y el ORM Propel) lo que permitió el desarrollo de la solución propuesta para el Sistema de Gestión para la Ingeniería Clínica y Electromedicina.

- ☑ Se personalizó un documento XSL que realiza transformaciones que tienen como resultado un YML optimizado para Propel en función de las pautas del modelo de datos utilizado por los desarrolladores del Sistema de Gestión para la Ingeniería Clínica y Electromedicina.

- ☑ Se desarrolló una aplicación que permite transformar un documento XML desarrollado en el DB Designer Fork, en un esquema YML optimizado para Propel, la misma facilita y optimiza el proceso de migración de un SGBD a otro.

- ☑ Se validó la solución propuesta para el ORM Propel y se entregó la documentación de la solución propuesta.

Recomendaciones

- ☑ Utilizar la solución propuesta en los demás sistemas del Departamento de Sistemas de Apoyo a la Salud (SAS) que precisen de sus funcionalidades, no únicamente en el Sistema de Gestión para la Ingeniera Clínica y Electromedicina (SIGICEM).
- ☑ Implementar una aplicación que permita transformar un documento XML dado en un esquema YML optimizado para Doctrine.

Referencias Bibliográficas

1. Orallo, José Hernández. La disciplina de los sistemas de Bases de Datos. *Historia, Situación Actual y Perspectivas.* [En línea] http://palomo.usach.cl/Docs/BD/JHernandezO-La_disciplinade_los_sistemas.pdf.

2. Marin, Massiel Guerra. *"SIGICEM: Análisis del módulo Gestión de Servicios Técnicos".* Ciudad de la Habana : s.n., 2009.

3. Silvente, Serguéi Frómeta. *Modelo lógico y físico de la base de datos del módulo de Investigaciones Forenses del proyecto CICPC.* Ciudad de la Habana : s.n., 2008.

5. Pérez, Sara. *Modelos de Base de Datos.* 2010.

7. Todo programas DB Designer Fork. [En línea] [Citado el: 17 de Noviembre de 2010.] http://www.todoprogramas.com/programa/dbdesignerfork..

8. NetBeans. [En línea] [Citado el: 10 de Diciembre de 2010.] http://blogultura.com/java/netbeans-6-8-liberado/.

9. The Apache Jakarta Project. *Apache Jmeter.* [En línea] [Citado el: 15 de Noviembre de 2010.] http://jakarta.apache.org/jmeter/.

10. [En línea] [Citado el: 22 de Febrero de 2011.] http://www.freedownloadmanager.org/es/downloads/Generador_de_Datos_de_DTM_22345_p/.

11. Maestros del Web. [En línea] [Citado el: 20 de Enero de 2010.] http://www.maestrosdelweb.com/editorial/el-framework-symfony-una-introduccion-practica-i-parte/.

12. Tecnoretales. *Introducción al ORM.* [En línea] [Citado el: 15 de Noviembre de 2010.] http://www.tecnoretales.com/programacion/..

13. Utilizando Doctrine como ORM en PHP. [En línea] [Citado el: 5 de Diciembre de 2010.] http://web.ontuts.com/tutoriales/utilizando-doctrine-como-orm-en-php/..

14. Propel. [En línea] [Citado el: 26 de Noviembre de 2010.] http://www.propelorm.org/.

15. utilizando doctrine como orm en php. [En línea] [Citado el: 16 de Febrero de 2011.] http://web.ontuts.com/tutoriales/utilizando-doctrine-como-orm-en-php/.

16. Centro de Estudiantes de Ingieneria de Sistemas. [En línea] [Citado el: 16 de Noviembre de 2010.] http://ceisuss.wordpress.com/2008/10/14/%C2%BFque-es-y-para-que-me-sirve-xml/.

17. Merelo, J.J. Generación de páginas Web usando XSLT y XML. [En línea] [Citado el: 10 de Diciembre de 2010.] http://geneura.ugr.es/~jmerelo/XSLT/.

18. cnieto. YAML. [En línea] [Citado el: 8 de Diciembre de 2010.] http://dotpress.wordpress.com/2007/10/16/yaml/.

19. Que es php. [En línea] [Citado el: 20 de Febrero de 2011.] http://www.maestrosdelweb.com/editorial/phpintro/.

20. Zaninotto, François Fabien Potencier. *Symfony la guía definitiva.*

21. ÍDEM 16.

22. Plugins. [En línea] [Citado el: 25 de Febrero de 2011.] http://www.librosweb.es/symfony/capitulo17/plugins.html.

23. Ballester, Lidier González. *Diseño de una base de datos para el control de los RRHH en los polos productivos de la facultad 9.* 2009.

24. Alfonso, Yoelvis Pozo. *Diseño del modelo de datos del Sistema de Colaboración Médica.* Ciudad de La Habana : s.n., 2010.

25. Macías, Dayron Fernández. *Diseño de una base de datos para controlar la información de Unión de Jóvenes Comunistas en la UCI.* 2008.

26. Microsoft. Integridad de los datos. [En línea] [Citado el: 5 de Marzo de 2011.] http://msdn.microsoft.com/es-es/library/ms184276.aspx.

27. sfDB4toPropelPlugin. [En línea] [Citado el: 22 de Noviembre de 2010.] http://www.strangebuzz.com/index.php/2008/09/08/36-new-symfony-11-plugin-tutorial-sfdb4topropelplugin.

Bibliografía

- ☑ BIBLIOGRAPHY [En línea]. - 22 de Febrero de 2011. - http://www.freedownloadmanager.org/es/downloads/Generador_de_Datos_de_DTM_22345_p/.
- ☑ Alfonso Pozo Yoelvis Diseño del modelo de datos del Sistema de Colaboración Médica [Libro]. - Ciudad de La Habana : [s.n.], 2010.
- ☑ Ballester González Lidier Diseño de una base de datos para el control de los RRHH en los polos productivos de la facultad 9 [Libro]. - 2009.
- ☑ Centro de documentación [En línea] // XSLT. - 15 de Marzo de 2011. - https://developer.mozilla.org/es/XSLT.
- ☑ Centro de Estudiantes de Ingieneria de Sistemas [En línea]. - 16 de Noviembre de 2010. - http://ceisuss.wordpress.com/2008/10/14/%C2%BFque-es-y-para-que-me-sirve-xml/.
- ☑ cnieto YAML [En línea]. - 8 de Diciembre de 2010. - http://dotpress.wordpress.com/2007/10/16/yaml/.
- ☑ DesarrolloWeb.com [En línea] // Introducción a XML. - 6 de Febrero de 22. - http://www.desarrolloweb.com/manuales/18/.
- ☑ DesarrolloWeb.com [En línea] // PHP a fondo. - 15 de Enero de 2011. - http://www.desarrolloweb.com/php/.
- ☑ Guia de JMeter [En línea]. - 15 de Diciembre de 2011. - http://www.osmosislatina.com/jmeter/pruebabasica.htm.
- ☑ Introducción al lenguaje XML [En línea]. - 19 de Abril de 2011. - http://geneura.ugr.es/~jmerelo/xml/.
- ☑ JMeter [En línea]. - 10 de Noviembre de 2011. - http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=jmeter.
- ☑ Macías Fernández Dayron Diseño de una base de datos para controlar la información de Unión de Jóvenes Comunistas en la UCI [Libro]. - 2008.
- ☑ Maestros del Web [En línea]. - 20 de Enero de 2010. - http://www.maestrosdelweb.com/editorial/el-framework-symfony-una-introduccion-practica-i-parte/.
- ☑ Marin Guerra Massiel “SIGICEM: Análisis del módulo Gestión de Servicios Técnicos”. [Libro]. - Ciudad de la Habana : [s.n.], 2009.

- ☑ Merelo J.J Generación de páginas Web usando XSLT y XML [En línea]. - 10 de Diciembre de 2010. - http://geneura.ugr.es/~jmerelo/XSLT/.
- ☑ Microsoft Integridad de los datos [En línea]. - 5 de Marzo de 2011. - http://msdn.microsoft.com/es-es/library/ms184276.aspx.
- ☑ NetBeans. [En línea]. - 10 de Diciembre de 2010. - http://blogultura.com/java/netbeans-6-8-liberado/.
- ☑ Orallo Hernández José La disciplina de los sistemas de Bases de Datos [En línea] // Historia, Situación Actual y Perspectivas. - http://palomo.usach.cl/Docs/BD/JHernandezO-La_disciplinade_los_sistemas.pdf.
- ☑ Pérez Sara Modelos de Base de Datos [Libro]. - 2010.
- ☑ Plugins [En línea]. - 25 de Febrero de 2011. - http://www.librosweb.es/symfony/capitulo17/plugins.html.
- ☑ Propel [En línea]. - 26 de Noviembre de 2010. - http://www.propelorm.org/.
- ☑ Que es php [En línea]. - 20 de Febrero de 2011. - http://www.maestrosdelweb.com/editorial/phpintro/.
- ☑ sfDB4toPropelPlugin [En línea]. - 22 de Noviembre de 2010. - http://www.strangebuzz.com/index.php/2008/09/08/36-new-symfony-11-plugin-tutorial-sfdb4topropelplugin.
- ☑ Silvente Frómeta Serguéi Modelo lógico y físico de la base de datos del módulo de Investigaciones Forenses del proyecto CICPC [Libro]. - Ciudad de la Habana : [s.n.], 2008.
- ☑ Tecnoretales [En línea] // Introducción al ORM. - 15 de Noviembre de 2010. - http://www.tecnoretales.com/programacion/..
- ☑ The Apache Jakarta Project [En línea] // Apache Jmeter. - 15 de Noviembre de 2010. - http://jakarta.apache.org/jmeter/.
- ☑ Todo programas DB Designer Fork [En línea]. - 17 de Noviembre de 2010. - http://www.todoprogramas.com/programa/dbdesignerfork..
- ☑ Uso de XSL para transformar la factura electronica XML [En línea]. - 20 de Enero de 2011. - http://www.lacorona.com.mx/fortiz/sat/xsl.php.
- ☑ utilizando doctrine como orm en php [En línea]. - 16 de Febrero de 2011. - http://web.ontuts.com/tutoriales/utilizando-doctrine-como-orm-en-php/.

- ☑ Utilizando Doctrine como ORM en PHP [En línea]. - 5 de Diciembre de 2010. - http://web.ontuts.com/tutoriales/utilizando-doctrine-como-orm-en-php/..
- ☑ Web Estilo [En línea] // Manual de PHP. - 25 de Abril de 2011. - http://www.webestilo.com/php/.
- ☑ Zaninotto François Fabien Potencier Symfony la guía definitiva [Libro].

Glosario de Términos

- ☑ **Estandarización:** Redacción y aprobación de normas que se establecen para garantizar el acoplamiento de elementos construidos independientemente así como garantizar la calidad de los elementos fabricados la seguridad de funcionamiento para trabajar con responsabilidad social.

- ☑ **Framework:** En el desarrollo de software, es una estructura de soporte definida, mediante la cual otro proyecto de software puede ser organizado y desarrollado.

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

- ☑ **Modelo de datos:** Lógica de negocio de las aplicaciones web, donde se encuentran las Bases de Datos.

- ☑ **MySQL:** Sistema Gestor de Bases de datos.

- ☑ **ORM**: (Mapeo de Objeto Relacional), que utiliza técnicas de programación para convertir datos a objetos y viceversa, permitiendo el trabajo con datos persistentes como si formaran parte de una BD orientada a objetos.

- ☑ **Plugin:** Aplicación que se relaciona con otra para aportarle una función nueva y generalmente muy específica.

- ☑ **PostgreSQL:** Sistema Gestor de Bases de datos.

- ☑ **Propel:** Capa de abstracción que se utiliza como ORM. Proporciona persistencia para los objetos y un servicio de consultas optimizadas.

- ☑ **Pruebas de carga intensiva:** Determina y asegura que el sistema funciona apropiadamente aún más allá de la carga de trabajo máxima esperada.

Anexos

```
nombrePlugin/
  config/
    *schema.yml         // Esquema de datos
    *schema.xml
    config.php          // Configuración específica del plugin
  data/
    generator/
      sfPropelAdmin
        */              // Temas para el generador de administraciones
          template/
          skeleton/
    fixtures/
      *.yml             // Archivos de fixtures
  lib/
    *.php               // Clases
    helper/
      *.php             // Helpers
    model/

      *.php               // Clases del modelo
    task/
      *Task.class.php   // Tareas de la línea de comandos
  modules/
    */                    // Módulos
      actions/
        actions.class.php
      config/
        module.yml
        view.yml
        security.yml
      templates/
        *.php
      validate/
        *.yml
  web/
    *                     // Archivos estáticos
```

Anexo 1: Estructura de archivos de un plugin

MySQL

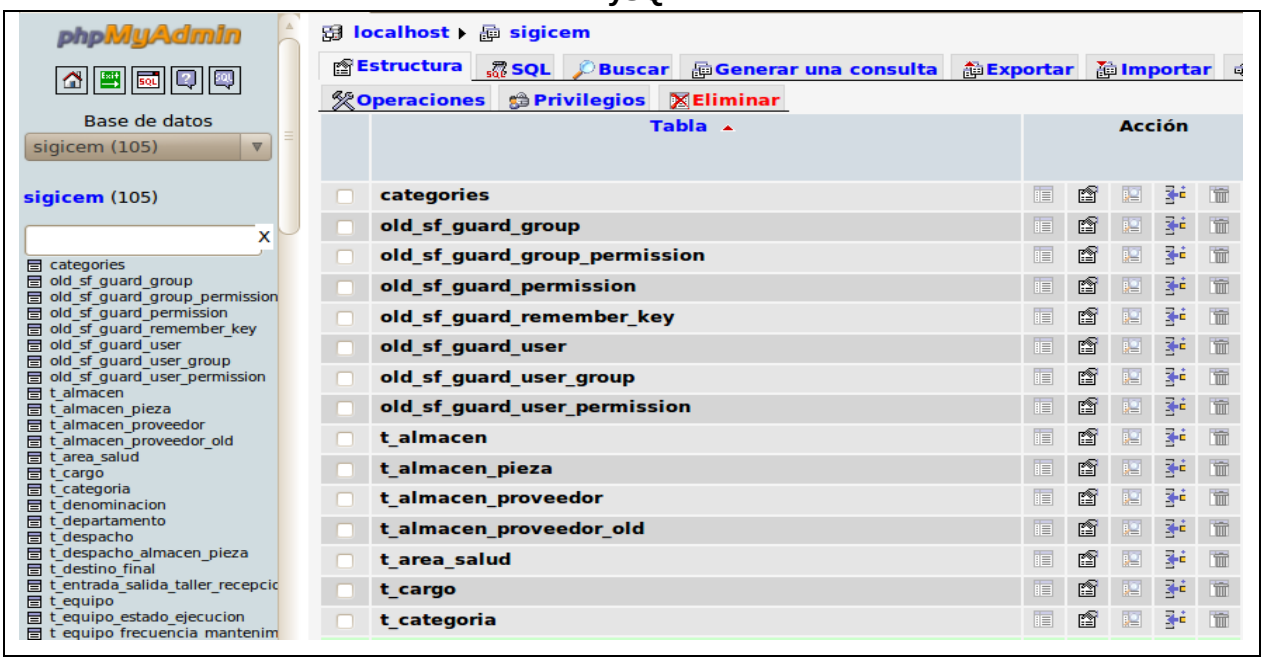
```
all:
  propel:
    class: sfPropelDatabase
    param:
      dsn: mysql:dbname=sigicem;host=localhost
      username: root
      password: root
      encoding: utf8
      persistent: true
      pooling: true
```

PostgreSQL

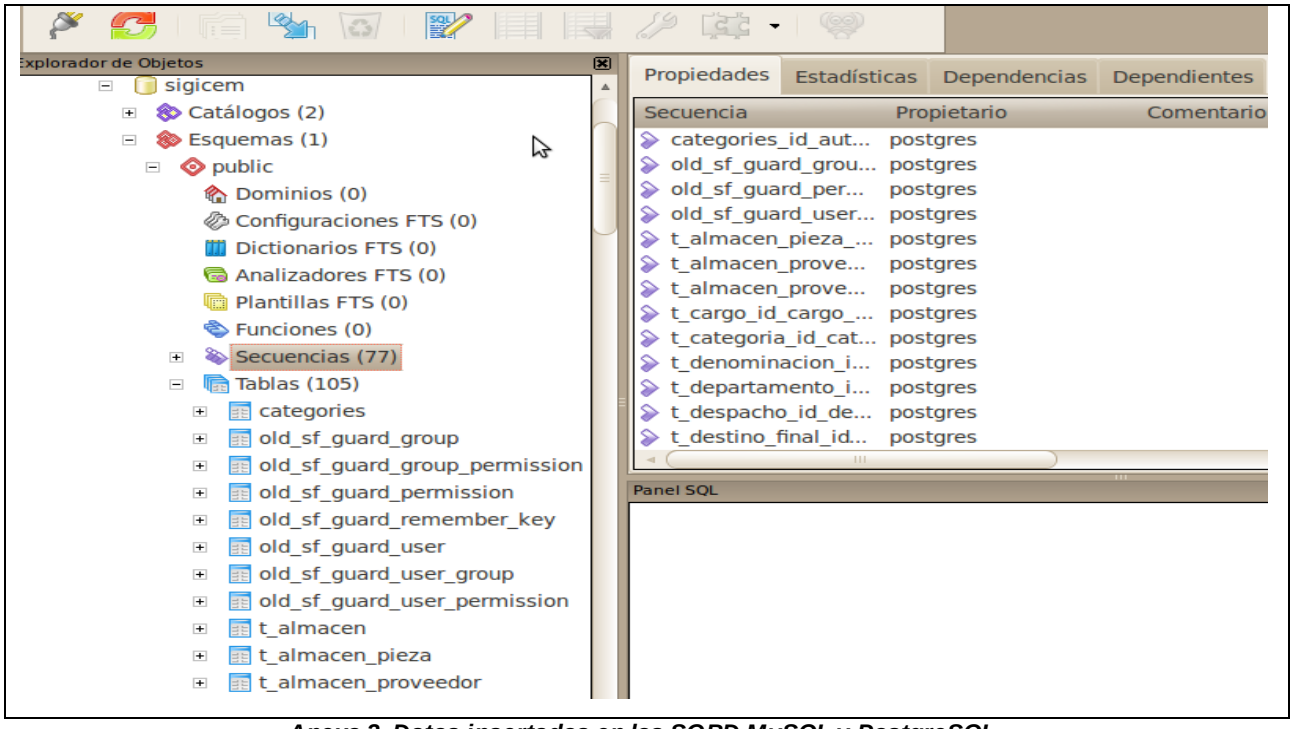
```
all:
  propel:
    class: sfPropelDatabase
    param:
      dsn: pgsql:dbname=sigicem;host=localhost
      username: postgres
      password: root
      encoding: utf8
      persistent: true
      pooling: true
```

Anexo 1: Configuración del archivo Database.yml utilizado para la conexión son los SGBD MySQL y Postgres SQL.

MySQL



PostgreSQL



Anexo 2. Datos insertados en los SGBD MySQL y PostgreSQL.

MySQL

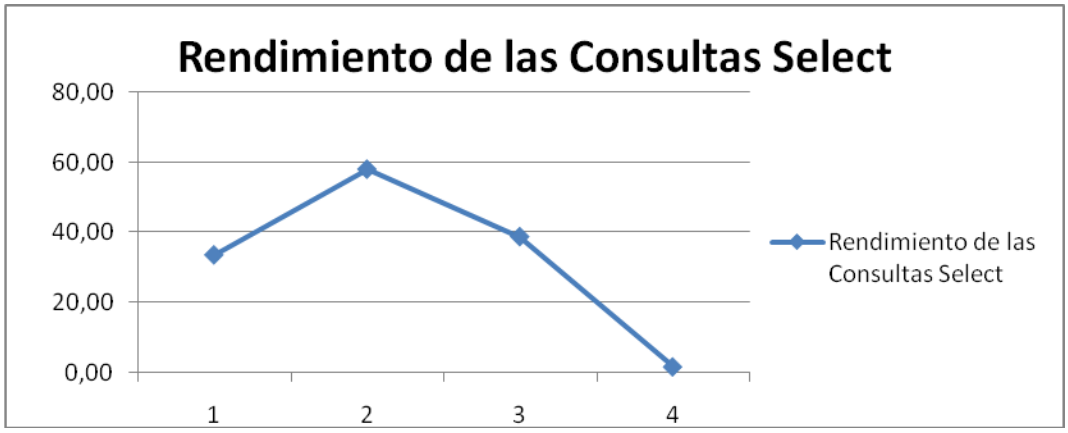
# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento
10	204	219	281	141	281	0,00%	33,7/sec
10	85	93	140	31	140	0,00%	58,1/sec
10	9146	9578	11531	5937	11531	0,00%	38,9/min
10	5503	5515	5531	5469	5531	0,00%	1,8/sec
40	3735	5469	9578	31	11531	0,00%	52,5/min

PostgreSQL

# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento
10	2448	1640	4141	1125	4141	0,00%	2,4/sec
10	1001	985	1110	969	1110	0,00%	8,5/sec
10	7482	7453	7734	7297	7734	0,00%	1,3/sec
10	771	781	812	734	812	0,00%	12,3/sec
40	2926	1125	7453	734	7734	0,00%	59,2/min

Anexo 3: Resultado de las consultas de selección en ambos SGBD.

MySQL



PostgreSQL

Anexo 4 Comportamiento del rendimiento para las consultas de selección en ambos SGBD.

MySQL

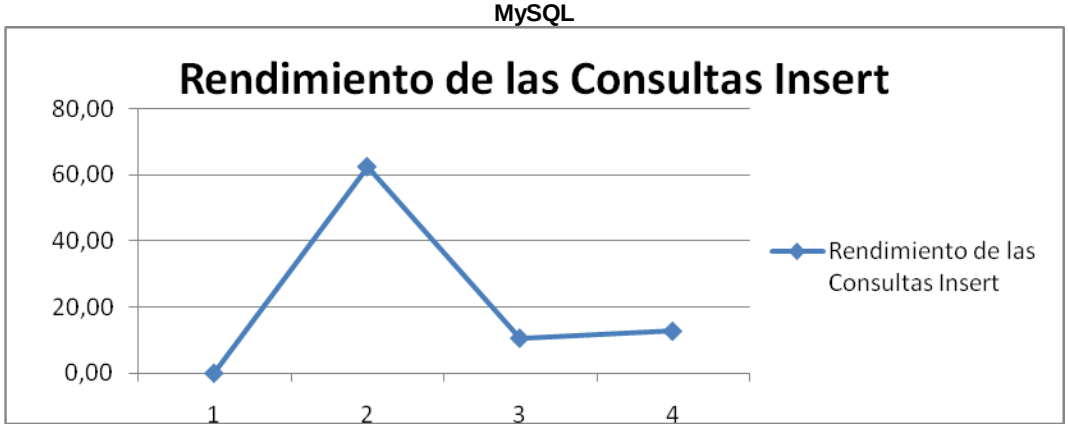
# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento
1	0	0	0	0	0	0,00%	∞/sec
1	16	16	16	16	16	0,00%	62,5/sec
1	94	94	94	94	94	0,00%	10,6/sec
1	79	79	79	79	79	0,00%	12,7/sec

PostgreSQL

# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento
1	0	0	0	0	0	0,00%	∞/sec
1	16	16	16	16	16	0,00%	62,5/sec
1	0	0	0	0	0	0,00%	∞/sec
1	0	0	0	0	0	0,00%	∞/sec

Anexo 5: Resultado de las consultas de inserción en ambos SGBD.

MySQL



PostgreSQL

Anexo 6 Comportamiento del rendimiento para las consultas de inserción en ambos SGBD.

MySQL

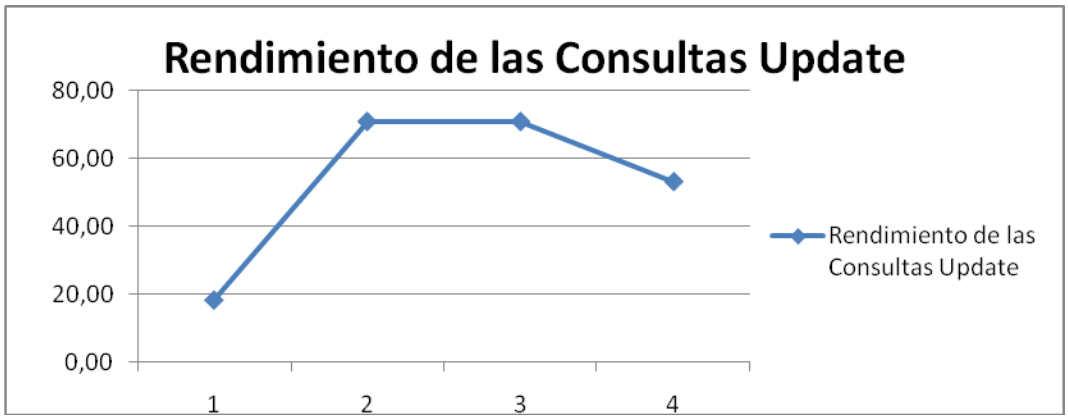
# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento
10	537	531	547	531	547	0,00%	18,3/sec
10	128	125	141	125	141	0,00%	70,9/sec
10	137	141	141	125	141	0,00%	70,9/sec
10	184	188	188	172	188	0,00%	53,2/sec

PostgreSQL

# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento
10	876	875	953	828	953	0,00%	10,3/sec
10	492	500	500	485	500	0,00%	20,0/sec
10	470	469	485	469	485	0,00%	20,6/sec
10	461	469	469	453	469	0,00%	21,3/sec
40	575	485	875	453	953	0,00%	1,2/sec

Anexo 7: Resultado de las consultas UPDATES en ambos SGBD.

MySQL



PostgreSQL

Anexo 8 Comportamiento del rendimiento para las consultas de actualización en ambos SGBD.

#	Consultas de Selección.
1	SELECT t_pieza.id_pieza, t_pieza.id_nomenclador_equipo, t_pieza.id_unidad_medida, t_pieza.descripcion, t_pieza.ref_fab, t_pieza.gastable, t_pieza.standar, t_pieza.precio_unitario, t_pieza.codigo FROM t_pieza

2	SELECT t_especialista.id_especialista, t_especialista.id_grupo_tecnico_brigada, t_especialista.id_especialidad_atiende, t_especialista.id_municipio, t_especialista.id_cargo, t_especialista.nombre, t_especialista.apellido1, t_especialista.apellido2, t_especialista.ci FROM t_especialista
3	SELECT ***** FROM public.t_especialista INNER JOIN public.t_especialista_atiende ON (public.t_especialista.id_especialista = public.t_especialista_atiende. Id_especialista) INNER JOIN public.t_especialista_graduado_de ON (public.t_especialista.id_especialista = public.t_especialista_graduado_de.id_especialista) WHERE public.t_especialista_atiende.id_especialista = public.t_especialista.id_especialista ORDER BY public.t_especialista.id_cargo
4	SELECT t_pieza.descripcion FROMt_pieza INNER JOIN t_pieza_ref_fab ON (t_pieza.id_pieza = pieza_ref_fab.id_pieza_ref_fab) WHERE (t_pieza.precio_unitario< '5144,90')

#	**Consultas de Inserción.**
1	INSERT INTO t_tipo_unidad_salud (id_tipo_unidad_salud, tipo_unidad) VALUES ('512096690','hospital')
2	INSERT INTO t_tipo_unidad_salud (id_tipo_unidad_salud, tipo_unidad) VALUES ('7106690','hospital')
3	INSERT INTO t_fabricante (id_fabricante, fabricante) VALUES (6166018,'cimex')
4	INSERT INTO t_equipo_estado_ejecucion (id_equipo_estado_ejecucion, estado, descripcion) VALUES (‘1918005’, '0619598', '022364')

#	Consultas de Actualización.
1	UPDATE t_pieza SET descripcion = 'libre' WHERE t_pieza.id_pieza = 5
2	UPDATE public.t_nomenclador_equipo SET codigo = 'CL' WHERE public.t_nomenclador_equipo.id_nomenclador_equipo = 5
3	UPDATE public.t_orden_servicio SET id_reporte_equipo = 'JG' WHERE public.t_orden_servicio.id_orden_servicio = 8
4	UPDATE public.t_equipo SET id_unidad_salud = 'ISO' WHERE public.t_equipo.id_equipo = '6'

Anexo 7: Consultas seleccionadas para la realización de las pruebas.