Análisis y Diseño de la versión 2.0 del módulo Réplica de la Plataforma Educativa Zera

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

Autora: Miroslava Lazara Aldana Cuza

Tutores: Ing. Enrique José Altuna Castillo

Ing. Yuleisy González Pérez

Cotutor: MSc. Roberto López Dosagües

La Habana 2012

Declaratoria de Autoría

Yo, Miroslava Lazara Aldana Cuza, declaro ser la autora de la presente tesis, y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma con carácter exclusivo.

Para que así conste, firmo la presente a los ____días del mes de _____del año 2012.

_____

Firma del Autor
Miroslava Lazara Aldana Cuza

_____ _____

Firma del Tutor
Ing. Enrique José Altuna Castillo

Firma del Tutor
Ing. Yuleisy González Pérez

_____

Firma del Cotutor
MSc. Roberto López Dosagües

AGRADECIMIENTOS

Gracias a Dios por permitirme llegar hasta aquí al igual que a mi familia.

A mi mamá que es la mejor en el mundo entero, por quererme tanto, apoyarme siempre e inculcar en mí sus sentimientos. Gracias mami por hacerme la persona que soy hoy.

A mis más sinceras amigas, mis hermanas:

Massiel: gracias por guiarme, por poner en mí toda tu confianza, por preocuparte y ayudarme en todo momento, gracias por estar pendiente de mí prácticamente todos los días durante 5 años.

Tania: gracias por todo el apoyo dado, por cuidarme, por tu comprensión y brindarme buenos consejos. Gracias por quererme tanto.

A mis cuñados, pues gracias a ellos en lugar de un padre tengo dos, los quiero mucho.

Al amor de mi vida por apoyarme, ayudarme y cuidarme durante estos 6 años, gracias por quererme. Te amo.

A mis tutores:

Yuleisy: por ser la persona más insistente que he conocido, por preocuparse y ayudarme para lograr satisfactoriamente la culminación del trabajo.

Enrique: por aclarar mis dudas y ayudarme cuando lo necesitaba.

A Dosagües por ser más que mi cotutor un amigo, por preocuparse, brindarme su apoyo y ayuda incondicional.

A todas las personas que me ayudaron especialmente a mis amigos Dairy, Danay, Yusdel y Mario.

Al tribunal de tesis por las recomendaciones dadas, en especial a Yanirys por las críticas constructivas cada vez que lo necesité.

Al Decano Pedro Luis Basulto quien se convirtió en un amigo y ejemplo a seguir. Gracias por todos los consejos dados y el apoyo brindado.

A la familia que se nos permite elegir, a todos mis amigos, con los que he disfrutado y sufrido esta etapa de mi vida.

A la persona que sin tener mi sangre se convirtió en mi hermana Nani, gracias por cuidarme cuando estuve enferma, por soportar todas mis malcriadeces y por estar conmigo en las buenas y en las malas.

A todas las personas que me cuidaron y apoyaron cuando estuve en Venezuela, Yusi, Yamira, Braulio, Yamila, Mileidis, Leidis, Tere y Juan.

A mi mamá y mis hermanas por ser las estrellas que iluminan mi camino.

Resumen

La Universidad de las Ciencias Informáticas (UCI) es un centro de educación superior en la que se desarrollan numerosos proyectos productivos. Entre estos en el Centro de Tecnologías para la Formación se desarrolla el proyecto Plataforma Educativa Zera. La Plataforma Educativa Zera gestiona y administra información relacionada con los estudiantes, profesores y contenidos educativos de varias instituciones. La cual cuenta con una o varias bases de datos locales que se encuentran asociadas a la base de datos central. Para el correcto funcionamiento y actualización del flujo de datos en el sistema, se realizan réplicas y sincronizaciones entre la base de datos central y las locales. Estos procesos son manejados por el módulo Réplica, el cual presenta varios problemas de funcionamiento. La presente investigación tiene como propósito desarrollar el análisis y diseño de la versión 2.0 del módulo Réplica de la Plataforma Educativa Zera. Además, validar la propuesta presentada como garantía de la calidad de su futura implementación.

Para ello se realizó un estudio sobre los conceptos asociados a los procesos de replicación de datos, las características de soluciones similares a nivel nacional e internacional, las herramientas de replicación, las metodologías de desarrollo de software, los lenguajes y herramientas de modelado. También, se investiga acerca de los lenguajes de programación, *framework* de desarrollo y sistema gestor de base de datos definidos en la arquitectura de la Plataforma Educativa Zera.

La propuesta brinda una solución a los problemas de réplica de datos de la Plataforma Educativa Zera. En la que son especificadas las funcionalidades que deberá tener el módulo Réplica y lo que debe implementarse para lograr el correcto funcionamiento del mismo.

Palabras claves: nodo, réplica, sincronización.

Índice

Introducción

Las Tecnologías de la Información y las Comunicaciones (TIC) se han desarrollado a un ritmo acelerado empleándose en todos los ámbitos de la sociedad. Con la digitalización de la enseñanza, el saber y el conocimiento son productos de una inteligencia colectiva, que transformada constantemente deja de tener residencia en las escuelas y otros centros educacionales para situarse en cualquier punto de la red. La educación o aprendizaje electrónico (*e-Learning*) es el suministro de programas educacionales y sistemas de aprendizaje a través de medios electrónicos. (1) Establece un nuevo modo de sociabilidad entre profesores y estudiantes donde el saber no se acumula ni se desgasta, sino que se convierte en un flujo de constante enriquecimiento y permanente circulación. La misma se basa en la utilización de una computadora u otro dispositivo electrónico para proveer a las personas de material educativo.

La educación electrónica unida al creciente perfeccionamiento de las TIC ha dado auge al desarrollo de sistemas de aprendizaje electrónico. En estos sistemas, los datos a los cuales acceden los profesores y estudiantes deben proveerse de modo sencillo y eficiente por medio de aplicaciones que actúen como puente entre estos y la información. Por tanto, se hace imprescindible garantizar que las tecnologías y herramientas utilizadas en el desarrollo de los sistemas de aprendizaje electrónico sean las apropiadas.

En un sistema informático los datos no deben estar centralizados, con lo que se evita que frente a algún fallo, se afecte la disponibilidad o la perdida de la información. Por lo cual es necesario almacenarlos en diferentes puntos en la red. Para lograr la descentralización de los datos, estos son almacenados en Bases de Datos (BD) locales conectadas a través de la red, formando un Sistema de Base de Datos Distribuidas (SBDD), en el cual se puede acceder a las BD locales más que a una BD central. Con el objetivo de mantener sincronizados los datos entre las BD del SBDD es utilizada la técnica de replicación de datos. (2) La replicación es la operación de transportar idénticamente los datos de las tablas de una BD hacia otra BD a través de la red, posibilitando la corrección, disponibilidad, coordinación y efectividad de los datos.

Los sistemas de aprendizaje electrónico trabajan en red, siendo capaces de ser actualizados, almacenados, recuperados, distribuidos y permitiendo compartir instrucciones o información. (3) En ellos se evidencia el apoyo a la educación de los SBDD, pues facilitan el acceso a la información y el intercambio entre profesores y alumnos de forma no presencial.

Cuba no queda ajena ante el desarrollo y aplicación del aprendizaje electrónico, integrándolo como nuevo método de enseñanza-aprendizaje a sus centros educacionales, un ejemplo de ello es la UCI. En la misma se desarrollan sistemas para la gestión del aprendizaje. Entre los numerosos proyectos desarrollados en la universidad, existen los enfocados al perfeccionamiento de las tecnologías informáticas en función del aprendizaje asistido por computadoras. Específicamente en el Centro de Tecnologías para la Formación (FORTES), se desarrolla la Plataforma Educativa Zera. La cual gestiona y administra las escuelas asociadas a la misma, así como los datos referentes a sus estudiantes, profesores y contenidos educativos.

En la Plataforma Educativa Zera no era posible asegurar la conectividad constante de todas las escuelas asociadas a la misma, por lo cual se incluyeron subconjuntos de la plataforma encada una de estas instituciones educativas, de forma tal que pudiesen trabajar sin conexión con el servidor central en caso de fallos en la conexión. De esta manera las BD de la Plataforma Educativa Zera forman un SBDD.

La Plataforma Educativa Zera cuenta con una BD central y una local por cada escuela asociada a ella. Las BD locales almacenan única y específicamente la información referente a la escuela correspondiente y por otra parte, la BD central recopila y almacena los datos pertenecientes a estas BD locales. Con esta arquitectura y la existencia de coherencia y actualización entre los datos de la BD central y las locales, se puede acceder a la Plataforma Educativa Zera desde la escuela o cualquier otro sitio en la red. Las entradas desde la escuela son procesadas en el servidor local y las entradas desde cualquier otro punto en la red se procesan en el servidor central. Para que exista un correcto funcionamiento, los datos entre las BD locales y la BD central deben permanecer actualizados y coherentes, gestión que dentro de la Plataforma Educativa Zera debe ser realizada a través del módulo Réplica.

El módulo Réplica surge durante la explotación de la Plataforma Educativa Zera, pues ante la inclusión de nuevas escuelas aparece la necesidad de configurar la replicación de sus datos. Este módulo ofrece una interfaz que posibilita la administración y configuración de los procesos de réplica de datos en la Plataforma Educativa Zera. Actualmente no garantiza que la información esté actualizada y coherente entre los servidores de BD, pues no completa los procesos de replicación, afectando la integridad, disponibilidad y confiabilidad de la información procesada en la Plataforma Educativa Zera.

Las deficiencias en el funcionamiento del módulo están dadas por la aparición de continuos errores durante los intentos de sincronización de los datos entre los servidores de BD locales y el central. Algunos de los cuales se presentan a continuación:

Los cambios realizados en la estructura de las tablas de la BD central estos no son sincronizados en los servidores locales, impidiendo la replicación de los datos entre el servidor de BD central y los locales.

Al ocurrir fallos en la conexión de algún servidor de BD, la Plataforma Educativa Zera continúa su funcionamiento, al ser restablecida la conexión las BD automáticamente intentan sincronizarse, acción que es interrumpida por errores generados durante la ejecución de la sincronización.

Cuando se intentan replicar las tablas de la BD que tienen como precedencias otras tablas de la misma, el proceso de replicación falla, por tanto no existe una jerarquía en el orden en que se replican las tablas.

A partir del análisis antes expuesto, se define como **problema a resolver** ¿Cómo contribuir al perfeccionamiento del módulo Réplica de la Plataforma Educativa Zera?

Para darle solución a esta problemática se define como **objetivo general:** desarrollar el análisis y diseño de la versión 2.0 del módulo Réplica de la Plataforma Educativa Zera, definiendo como **objeto de estudio** los procesos de réplica de datos entre servidores de BD, teniendo como **campo de acción** los procesos de réplica de datos entre los servidores de BD de la Plataforma Educativa Zera.

La presente investigación estará sustentada en la siguiente **idea a defender**: si se desarrolla el análisis y diseño de la versión 2.0 del módulo Réplica de la Plataforma Educativa Zera, se contribuirá al perfeccionamiento del mismo.

Son definidos para esta investigación los siguientes **objetivos específicos:**

- Elaborar el marco teórico de la investigación acerca de las principales tendencias de replicación de datos.
- Definir el análisis de la versión 2.0 del módulo Réplica de la Plataforma Educativa Zera.
- Diseñar la versión 2.0 del módulo Réplica de la Plataforma Educativa Zera.
- Validar el análisis y diseño de la versión 2.0 del módulo Réplica de la Plataforma Educativa Zera.

Con el fin de lograr los objetivos propuestos se plantean las siguientes **tareas de investigación:**

- Elaboración del estado del arte y los fundamentos teóricos de los procesos de réplica de datos entre servidores de BD.

- Caracterización de la versión existente del módulo Réplica de la Plataforma Educativa Zera.
- Selección de las tecnologías, metodologías y herramientas adecuadas para el diseño de la versión 2.0 del módulo Réplica de la Plataforma Educativa Zera.
- Descripción de la versión 2.0 del módulo Réplica de la Plataforma Educativa Zera.
- Elaboración del Modelo de Dominio del sistema.
- Identificación de las mejoras del módulo, especificándolas en requerimientos funcionales y no funcionales para la versión 2.0 del módulo Réplica de la Plataforma Educativa Zera.
- Especificación de los casos de uso del sistema.
- Modelación de los Diagramas de Clases del Análisis, Diagrama de Colaboración del Análisis y los Diagramas de Clases del Diseño empleando estereotipos web.
- Definición del Modelo de Datos.
- Validación de la especificación de los requerimientos y de las clases del diseño.

El análisis y estudio de la parte de la ciencia que está siendo objeto de investigación, se realiza a través del empleo de la combinación de los métodos de investigación Teóricos y Empíricos. Entre los métodos de investigación Teóricos fueron empleados:

- **Analítico – Sintético*:*** a partir de un análisis de la bibliografía consultada, permitió la comprensión de los procesos de réplica y sincronización de datos. Además, seleccionar y sintetizar las herramientas a emplear para el desarrollo de la nueva versión del módulo Réplica de la Plataforma Educativa Zera.
- **Histórico – Lógico:** el desarrollo de la investigación se realizó partiendo del análisis de la versión existente del módulo Réplica de la Plataforma Educativo Zera, que permitió obtener conocimientos sobre la trayectoria de esta aplicación. Se realizó un estudio de las investigaciones existentes y los resultados obtenidos por otros autores sobre estos procesos, lo que fue utilizado como de base para la selección de la metodología, los lenguajes y herramientas adecuadas para el desarrollo de la aplicación.

Entre los métodos de investigación Empíricos se destacan:

- **Observación*:*** posibilitó la observación de los procesos de réplica en el actual módulo, así como el desarrollo de los eventos y la interacción de los mismos durante el proceso de validación.
- **Revisión de documentos*:*** contribuyó a la elaboración del marco teórico conceptual y el desarrollo del diseño de la propuesta de la nueva versión del módulo Réplica de la Plataforma Educativa Zera.

Capítulo I: Fundamentación Teórica

Introducción

En el capítulo que se exhibe a continuación se realiza un análisis de la literatura científica que con anterioridad pudo haber abordado problemas y objetos de investigación similares a los de esta investigación. El propósito es determinar el nivel de conocimiento sobre el problema en cuestión y reflejar un conjunto de proposiciones teóricas, agrupadas en sub epígrafes. Al mismo tiempo se van definiendo, desde el punto de vista metodológico, los conceptos principales, tendencias, herramientas y metodologías que sustentan este estudio y sus propuestas.

1.1 Bases de datos

El incremento del uso de las nuevas tecnologías ha traído un crecimiento directamente proporcional al manejo de información y datos en empresas, escuelas y otras instituciones. Lo que ha provocado la necesidad de almacenar la información de forma digitalizada, para lo que son de primordial empleo las BD.

Se consultaron diferentes sitios referenciados para obtener la definición de BD, lo que permitió realizar la propuesta del concepto de BD para esta investigación.

"Una base de datos es una colección de información organizada de forma que un programa de ordenador pueda seleccionar rápidamente los fragmentos de datos que necesite. Una base de datos es un sistema de archivos electrónico." (4)

"Una base de datos es una recopilación de información relativa a un asunto o propósito particular, como el seguimiento de pedidos de clientes o el mantenimiento de una colección de música. Si la base de datos no está almacenada en un equipo o sólo están instaladas partes de la misma, puede que deba hacerse un seguimiento de la información procedente de varias fuentes, en orden a coordinar y organizar la base de datos." (5)

"Una base de datos o banco de datos es un conjunto de datos que pertenecen al mismo contexto almacenados sistemáticamente para su posterior uso." (6)

Partiendo de las definiciones de BD consultadas para la presente investigación es definida una BD como una colección de información organizada y almacenada en un ordenador que permite la búsqueda, control y selección de la misma, lo que facilita el acceso a la información por varias aplicaciones. Estos datos son organizados de forma independiente a su utilización, accesibles en tiempo real y compatible con usuarios concurrentes y sus respectivas peticiones de información.

Dentro de la Plataforma Educativa Zera los datos gestionados de estudiantes, profesores y contenidos educativos son almacenados en las BD locales y la central, con el propósito de tener el control y administración de los mismos. Cada institución cuenta con una BD local asociada a la BD central, compartiendo con esta un mismo esquema global, conformándose así un SBDD.

1.2 Sistema de base de datos distribuidos

Las innovaciones tecnológicas que han tenido lugar en los últimos años han promovido el desarrollo de los sistemas informáticos. Unido a esto se ha aumentado la necesidad de interactuar, integrar y compartir la información almacenada en las BD. Debido a la necesidad de facilitar el manejo de los datos almacenados en varios sitios conectados a través de la red, se ha propiciado el empleo de los SBDD.

Un SBDD de acuerdo a las bibliografías consultadas es definido por varios autores como:

"Una colección de informaciones o datos, que se encuentran ubicados en distintos puntos de la red y lógicamente organizados. Donde en cada uno de los puntos se tiene capacidad de procesamiento autónomo, pudiendo además, ejecutar aplicaciones locales." (7)

"En un sistema de base de datos distribuida, los datos se almacenan en varios computadores. Los computadores de un sistema distribuido se comunican entre sí a través de diversos medios de comunicación, tales como cables de alta velocidad o líneas telefónicas." (8)

"Son un grupo de datos que pertenecen a un sistema pero a su vez está repartido entre ordenadores de una misma red, ya sea a nivel local o cada uno en una diferente localización geográfica. Cada sitio en la red es autónomo en sus capacidades de procesamiento y es capaz de realizar operaciones locales y en cada uno de estos ordenadores. Debe estar ejecutándose una aplicación a nivel global que permita la consulta de todos los datos como si se tratase de uno solo." (9)

Los SBDD tienen múltiples ventajas, algunas de las cuales se mencionan a continuación: (10)

- Los datos son localizados en un lugar más cercano al de su utilización, por tanto, el acceso a estos es más rápido.
- El procesamiento es rápido debido a que varios nodos intervienen en el procesamiento de una carga de trabajo.
- Nuevos nodos se pueden agregar fácil y rápidamente.
- Los costos de operación se reducen.

- La probabilidad de que una falla en un solo nodo afecte al sistema es baja.
- Existe una autonomía e independencia entre los nodos.

Después del estudio realizado sobre los SBDD, se llegó a la conclusión de que un SBDD se define como: el conjunto de datos relacionados entre sí, que se encuentran distribuidos en distintos sitios interconectados a través de la red, donde las aplicaciones acceden a los datos desde distintos puntos de la misma. En estos, los usuarios de un sitio pueden acceder a sus datos desde otro sitio, pudiendo mantener los datos de forma local y también tener acceso a ellos de manera remota.

La información entre las BD de un SBDD debe mantenerse actualizada y coherente, para lo cual son empleadas varias técnicas de distribución y sincronización de los datos.

1.3 Réplica de datos

La replicación provee una forma más rápida y confiable de esparcir los datos entre las distintas BD en un sistema distribuido. Es definida por varios autores como:

"La replicación copia y mantiene los objetos de las bases de datos en las múltiples bases de datos que levantan un sistema distribuido. Puede mejorar el funcionamiento y proteger la disponibilidad de las aplicaciones porque alterna opciones de acceso a los datos existente." (2)

"La replicación de datos consiste en el transporte de datos entre dos o más servidores, permitiendo que ciertos datos de la base de datos estén almacenados en más de un sitio y así aumentar la disponibilidad de los datos y mejorar el rendimiento de las consultas globales." (11)

"El sistema conserva varias copias o réplicas idénticas de una tabla. Cada réplica se almacena en un nodo diferente." (9)

La replicación provee copias de seguridad, mejora el funcionamiento de las aplicaciones, garantiza la disponibilidad y reduce el acceso a los datos. Además, reduce el impacto producido por las caídas y el tráfico de la red y balancea la carga en los servidores centralizados, pues un sistema tiene acceso a una base de datos local más que al servidor central. Entre sus características se encuentran: (12)

Efectividad: depende de la forma en la que los datos sean distribuidos y almacenados. A mayor efectividad, mayor será la disponibilidad de datos para ejecutar procesos paralelos.

Alta Disponibilidad: es la razón de tiempo prudente en la que un servicio puede ser accedido. En el mejor de los casos puede ser de un 100%, a pesar de los fallos que

se puedan presentar en el servidor, ya que debe existir un servidor adicional que posea alguna técnica de replicación que lo pueda suplantar en caso de ser necesario.

Tolerancia a fallos: garantiza un comportamiento correcto, donde efectivamente pueden existir un número finito de fallos y tipos de fallos.

Coordinación: cada una de las partes que conforman la base de datos distribuida acuerda un consenso para realizar las invocaciones de los servicios a los objetos, que al final de la transacción debe realizarse tal y como fue solicitada, para lo cual debe utilizar algún tipo de ordenamiento.

Partiendo de lo anteriormente expuesto para esta investigación se define como réplica de datos: al proceso de repetición y distribución de las tablas de una BD entre diversas BD en un SBDD, permitiendo que los datos se encuentren disponibles en cualquier momento y desde cualquier sitio en la red. Es la acción de transportar la información de las tablas de la BD deseadas de un nodo hacia otro a través de la red.

Puede ser utilizada en dependencia de los requerimientos de los sistemas o instituciones que la vayan a emplear. Teniendo en cuenta el entorno, puede realizarse de dos formas:

- *Entorno maestro-maestro (par a par):* varios sitios en la red actúan como pares iguales, donde cada uno de ellos es maestro y se comunica con otro maestro. Permiten operaciones de lectura/escritura y cuando algún dato es agregado, modificado o eliminado, cada uno es actualizado, pudiéndose modificar el esquema de la BD de cualquiera de las réplicas de los sitios en la red, eliminando así la sobrecarga de un único sitio en la red. Este permite la réplica en ambos sentidos, o sea bidireccional. (13)
- *Entorno maestro-esclavo (solo lectura):* el maestro puede recibir consultas de lectura/escritura, mientras que los esclavos aceptan únicamente de lectura, por tanto, si se desea realizar algún cambio en la BD este debe realizarse en el maestro. La replicación es realizada en un único sentido. (14)

Del análisis realizado sobre los entornos de replicación planteados se percibe que el entorno maestro-maestro permite que los datos de un SBDD puedan ser actualizados por cualquier miembro del sistema. Además, es el responsable de la propagación de las modificaciones en los datos dentro del SBDD, posibilitando que la información viaje en varios sentidos. Por otra parte, en el entorno maestro-esclavo los datos solo pueden ser modificados por el servidor Maestro, restringiendo la replicación en un solo sentido, del maestro hacia los esclavos.

El correcto funcionamiento de un sistema distribuido implica la coherencia de los datos entre los sitios o servidores de BD, esto es logrado mediante la sincronización de datos entre ellos. La sincronización es el proceso de mantener análogos los datos entre los servidores de BD de un SBDD, a través del intercambio de los registros actualizados en cada sitio. Los modelos de sincronización y las técnicas de replicación utilizadas se clasifican en:

- *Réplica Asíncrona:* captura los cambios locales almacenándolos en una cola y a intervalos regulares y replica estos datos en los servidores remotos. Es más óptima para ambientes distribuidos donde los nodos estén a larga distancia unos de otros, mejorando el rendimiento de la aplicación y reduciendo el tráfico en la red. (15)
  - ✓ *Descarga y Recarga:* consiste en hacer una salva de los datos, copiar la misma para un dispositivo de almacenamiento para luego distribuirla por los demás servidores. Esta técnica presenta el inconveniente de que en la mayoría de las ocasiones se consultan datos que tienen semanas de estar desactualizados, además de que el proceso se realiza de forma manual. (16)
  - ✓ *Instantánea*: consiste en hacer una instantánea de una tabla en un momento dado y esta "foto" es la que se replica en las demás bases de datos. Aunque mucho más avanzada que la técnica de Descarga y Recarga, esta presenta el inconveniente de que las tablas esclavas son tablas de solo lectura. Se recomienda utilizar cuando la mayoría de los datos no cambian con frecuencia; se replican pequeñas cantidades de datos y cuando los sitios con frecuencia están desconectados y es aceptable un periodo de latencia largo (la cantidad de tiempo que transcurre entre la actualización de los datos en un sitio y en otro). (17)
  - ✓ *A través de disparadores (triggers):* la función del disparador es ejecutar una acción cuando ocurre un evento en la BD, los eventos pueden ser de inserción, actualización o eliminación. (18)
- *Réplica Síncrona:* llamada réplica en tiempo real, ejecuta cualquier procedimiento reproduciéndolo en todos los sitios del entorno de replicación como parte de una única transacción, si en algunos de los sitios ocurriera un fallo, la transacción entera es anulada. Eficiente en entornos distribuidos donde los nodos se encuentren a corta distancia unos de otros, pues esta incide en el rendimiento de la aplicación y aumenta el tráfico en la red. (15)

- ✓ *Confirmación en dos fases:* permite la sincronización de datos distribuidos. Cada transacción solamente es aceptada si todos los sistemas implicados en la réplica están conectados y listos para recibirla, si al menos uno falla, todo el proceso es anulado. (16)

Luego de analizar los distintos entornos y modelos para la réplica de datos, es seleccionado como entorno de replicación el maestro-maestro, pues a diferencia del maestro-esclavo, posibilita la replicación bidireccional entre el servidor central y los locales, minimizando el acceso al central, mejorando así el rendimiento de la aplicación. Para las sincronizaciones se decide utilizar el modelo asíncrono empleando la técnica de disparadores. Se llega a tal criterio teniendo en cuenta que los servidores de la Plataforma Educativa Zera quizás no tengan la misma ubicación geográfica. Este modelo mejorará el rendimiento y el tráfico en la red y mantendrá actualizadas las BD del sistema distribuido, pues en caso de que ocurra alguna falla en el servidor central esto no influirá en el funcionamiento y disponibilidad de la Plataforma Educativa Zera.

1.3.1 Conflictos en la replicación de datos

Los cambios dentro de una BD de un sistema distribuido pueden provocar conflictos de replicación. Al propagarse los datos de forma asíncrona y concurrentemente, puede que un mismo dato sea modificado por varios sitios en un instante de tiempo, generando así un conflicto, provocando que los datos no fluyan hasta que este sea resuelto. Por lo que es necesario tener en cuenta para el diseño de un sistema de replicación los posibles conflictos que puedan presentarse durante el proceso.

Conflicto de actualización:

> Cuando en más de un sitio intentan actualizar un mismo registro o dato concurrentemente, se produce un conflicto de actualización, siendo necesario determinar cuál de las actualizaciones se efectuará primero. (19)
>
> Para solucionar este conflicto existen varios mecanismos como son; cada servidor debe obtener una prioridad y el de mayor prioridad es el de mayor importancia ante los demás servidores; la más nueva o la más antigua de las modificaciones es la considerada correcta y por defecto, si no se eligió ninguno de los criterios gana la más nueva; se garantiza que cada registro sea manipulado por un único servidor, lo que simplifica la arquitectura. (20)

Conflicto de orden:

> Si la replicación a un sitio maestro A se encuentra bloqueada, la réplica de las modificaciones de los datos puede seguir ejecutándose a los demás sitios

> maestros, luego puede que estas modificaciones sean replicadas al sitio maestro A en un orden distinto al que ocurrieron, pudiendo producir un conflicto de orden. (19)
>
> Este tipo de conflicto suele resolverse asignándole distintas prioridades a los sitios maestros, de forma que se ordenen las transacciones de acuerdo a estas prioridades. (20)

Conflicto de unicidad:

> Si en dos transacciones desde dos sitios diferentes, cada una inserta una fila o registro en sus correspondientes tablas con valores iguales de clave primaria, esto provoca un conflicto de unicidad o carácter único. (19)
>
> Los mecanismo existentes para evitar este conflicto son; para cada servidor brindar un rango distinto de números para los generadores de claves; agregar el identificador del servidor a la clave primaria; replicar en tablas separadas y acceder a los datos a través de una vista formada por la unión de ellas. (20)

Conflicto de supresión:

> Cuando dos transacciones provenientes desde sitios diferentes, una intenta borrar un registro y otra actualizar o borrar este mismo registro. (19)
>
> Una posible solución es que los sitios marquen lógicamente los registros a ser borrados y que periódicamente el sitio maestro ejecuta un proceso que realice el "*delete*" físico de los datos, es decir que desde los sitios replicados no se pueda ejecutar una sentencia "*delete*". (20)

Del análisis de los conflictos de replicación, se considera que no es necesaria la implementación de estos mecanismos para la resolución de los conflictos de unicidad y supresión. Para el caso de los conflictos de unicidad los elementos de las tablas de la BD de la Plataforma Educativa Zera contienen identificadores únicos. Por otra parte, el método mencionado para la resolución de conflictos de supresión no debe ser implementado pues el entorno de replicación maestro-maestro plantea el funcionamiento de los servidores como pares iguales. Por tanto, se define que el módulo Réplica debe implementarlos mecanismos para evitar los conflictos de actualización y de orden. Ejecutando funciones internas que sean activadas ante la ocurrencia de un error durante la replicación.

1.4 Análisis de la actual versión del módulo Réplica

El objetivo del módulo Réplica de la Plataforma Educativa Zera es facilitar y garantizar la correcta administración de las réplicas y sincronizaciones entre los servidores de BD. El mismo basa su funcionamiento en la herramienta de replicación SymmetricDS 1.7.3,

siendo este módulo el puente entre el usuario y el SymmetricDS 1.7.3. Se desarrolló con la intención de garantizar la coherencia y actualización de los datos entre los servidores de BD locales y el central de la Plataforma Educativa Zera. Además, permitir que usuarios conocedores de los procesos de replicación y que no tengan conocimientos avanzados sobre el tema los administren.

Actualmente el módulo presenta problemas de funcionamiento, que impiden la correcta configuración de los procesos de replicación. No garantiza la sincronización entre los datos de la Plataforma Educativa Zera, lo que afecta en gran medida la utilidad de la misma.

A continuación se muestra un resumen de las deficiencias más significativas que ha presentado el módulo en su funcionamiento:

Frecuentemente surgen errores durante la replicación de tablas de la BD, impidiendo que este proceso sea completado. Un ejemplo de ello se evidencia cuando existe una relación de multiplicidad entre una tabla A y una B, es generada la tabla A_B si se intenta replicar la tabla A_B, sin antes ser replicadas A y B surgen conflictos que impiden la sincronización. Además, en el caso de una relación de uno (A) a muchos (B) donde B tiene una llave foránea de A, es necesario replicar primero A y luego B. Actualmente el módulo no cuenta con procedimientos para evitar estos errores.

Cuando es necesario realizar cambios estructurales a la BD, estos son efectuados en el servidor de BD central, siendo preciso la aplicación de estos cambios a las BD locales. Actualmente el módulo no cuenta con un procedimiento para realizar esta operación, provocando con esto la interrupción de los procesos de replicación de la Plataforma Educativa Zera. Lo que conlleva a seguir estrategias muy costosas, como mantener múltiples BD para las diferentes versiones de las escuelas, hasta que se pueda ir físicamente a realizar las modificaciones necesarias en este nodo.

Ocasionalmente surgen problemas de conexión o de funcionamiento en algunos de los servidores, en este caso la Plataforma Educativa Zera continúa funcionando y con ella las acciones sobre los datos. Estas acciones son capturadas y almacenadas para sincronizar sus datos automáticamente cuando se restablece la conexión entre los servidores BD. En este caso se producen errores que bloquean la replicación frente a los cuales el módulo Réplica no cuenta con métodos para darle solución.

A partir del análisis realizado se determina que el módulo Réplica de la Plataforma Educativa Zera carece de funcionalidades que permitan un efectivo funcionamiento de la

réplica de datos entre sus servidores de BD. No garantiza la actualización y coherencia entre los datos en los servidores de BD. Por tanto, se considera necesaria la investigación para desarrollar una nueva propuesta de solución a los problemas presentados por el módulo Réplica de la Plataforma Educativa Zera.

1.5 Análisis de otras soluciones existentes

A nivel nacional e internacional se han desarrollado varios sistemas para la réplica de datos que en su mayoría se apoyan en el funcionamiento de herramientas de replicación como son: PgCluster, Slony, Pgpool y PyReplica. Estas herramientas se basan generalmente en la configuración de ficheros manualmente por la persona encargada de la replicación.

Las soluciones existentes a nivel mundial son herramientas que integran el tratamiento de la réplica de datos en su totalidad. Son desarrolladas sobre software propietario y en su mayoría necesitan licencia para ser utilizadas.

En Cuba entre las soluciones propuestas para la réplica de datos se encuentran:

El sistema utilizado por la Empresa de las Telecomunicaciones de Cuba ETECSA: su arquitectura está compuesta por un servidor central y varios servidores locales, la información viaja en un solo sentido. Emplea el gestor de base de datos Oracle y realiza la replicación de los datos en tiempo real. (20)

La réplica propuesta por el Proyecto Identidad: se caracteriza por ser bidireccional, basada en el control de cambios y no en colas. Soporta el particionamiento horizontal de los datos y provee mecanismos de detección y resolución de conflictos. Está implementado sobre la plataforma .NET. (21)

Chronos: sistema de replicación asíncrona maestro-maestro para PostgreSQL: pretende satisfacer la mayor parte de los requerimientos de este tipo de soluciones. Ha sido diseñada a partir de las necesidades particulares de un escenario de réplica asumiendo las principales potencialidades de las soluciones similares en la actualidad para PostgreSQL. (14)

La réplica de BD del sistema SIGAC: es una propuesta que basa su funcionamiento en la herramienta de replicación Slony-I para PostgreSQL. Permite la replicación de datos a intervalos de tiempo y en una sola dirección. (20)

Del estudio realizado sobre el desarrollo de herramientas para la replicación de datos en Cuba, se destaca la aplicación de la réplica de datos a intervalos de tiempo, el tratamiento de errores y la resolución de conflictos. Estas herramientas aunque integran prácticamente

en su totalidad la gestión de los procesos de réplica de datos y resuelven problemas específicos de las entidades para las cuales fueron desarrolladas, no satisfacen las necesidades de la Plataforma Educativa Zera. De ellas Chronos e Identidad son las que más se acercan a las exigencias de la Plataforma Educativa Zera, pero no pueden ser embebidos dentro de ella. Por otra parte, el sistema de SIGAC y ETECSA no soportan la replicación en varios sentidos.

1.6 Herramientas de replicación

1.6.1 PgCluster

PgCluster es una herramienta para *clustering* de base de datos. Brinda la posibilidad de realizar la replicación síncrona y permite el entorno de replicación maestro-maestro. Está formado por tres tipos de servidores, un servidor para el balance de carga, un clúster de BD y un servidor de réplica. Cuando ocurre un fallo en el clúster de BD el servidor de balance de carga y el de replicación separan el fallo del sistema y continúa el servicio que usa la DB restante. La carga de la sesión de las demandas es distribuida. Es efectivo en aplicaciones Web donde existe gran demanda por el número de peticiones. (22)

Las ventajas más significativas para esta investigación presentadas por la herramienta de software libre PgCluster, es la aplicación del entorno de replicación maestro-maestro, no cuenta con un único punto de falla y permite el balanceo de carga. Entre las desventajas de utilización se encuentra el empleo del modelo síncrono, requiere altas prestaciones de hardware, su instalación y configuración es compleja. Debido a las desventajas antes mencionadas no es adecuada la utilización de PgCluster en la propuesta de solución para resolver los problemas presentados por el módulo Réplica.

1.6.2 PyReplica

PyReplica es una herramienta de replicación asíncrona. Soporta los dos entornos de replicación aunque en el caso del maestro-maestro su rendimiento es limitado. Permite la detección de conflictos, replicación condicional y notificaciones por correo. Además, el monitoreo de las conexiones sin el uso de protocolos especiales, ni herramientas externas y protegido con transacciones en dos fases. Está programado en Python por lo que es simple y flexible, permitiendo una fácil instalación. No es necesario aprender nuevos comandos para su administración y es muy fácil de adaptar manualmente. Es eficiente, pues tiene un bajo impacto en el uso de memoria y es multiplataforma. (23)

Entre sus característica se destaca el soporte de ambos ambientes de replicación, la detección de conflictos y el empleo del modelo de replicación asíncrono. A pesar de

cumplir con las características definidas para el desarrollo del módulo Réplica, no es la indicada pues presenta problemas de rendimiento en el empleo del entorno de replicación maestro-maestro.

1.6.3 SymmetricDS

SymmetricDS es un software de replicación de datos asíncrona que permite subscriptores múltiples y sincronización bidireccional. Utiliza tecnologías web y de BD para replicar tablas entre las BD relacionales asincrónicamente, casi en tiempo real. Fue diseñado para escalar a un gran número de BD, trabajar con conexiones de bajo ancho de banda, resistir a espacios de tiempo de inoperatividad de la red y se comporta como múltiples-maestros. Puede funcionar con los siguientes gestores de BD: MySQL, Oracle, SQL Server, PostgreSQL, DB2, Firebird, HSQLDB, H2, y Apache Derby. (24)

La aplicación SymmetricDS permite que los cambios de entrada y de salidas sean sincronizadas con otras BD. El nodo que inicia la conexión es el cliente y el nodo que la recibe es el servidor. Teniendo en cuenta que la sincronización está configurada para extraer o enviar en cualquier dirección, el mismo nodo puede actuar a la vez como cliente o como servidor en diferentes circunstancias.

SymmetricDS es independiente del gestor de base de datos que se utilice, siempre y cuando soporte la tecnología de *trigger* y *driver JDBC*[1]. Es una herramienta en constante evolución con una amplia comunidad, fácil de aprender y con más de una posibilidad para su despliegue, lo que permite usar la forma más adecuada según las características del hardware que se posea. (25)

Producto a su constante evolución se han desarrollado varias versiones del SymmetricDS que le han sumado funcionalidades y eliminado deficiencias, por lo que es conveniente valorar una versión más actual que la utilizada en el desarrollo del actual módulo Réplica de la Plataforma Educativa Zera.

1.6.2.1 SymmetricDS 1.7.3 vs SymmetricDS 2.4

SymmetricDS 2.4 está construido sobre la base del SymmetricDS 1.7.3 e incorpora varios cambios en su arquitectura y rendimiento. Ofrece nuevas funcionalidades que hacen más fácil el mapeo de las diferentes tablas durante la sincronización. Incluye nuevas opciones en la captura de los cambios que posibilitan la sincronización de objetos más grandes, que estaban previamente limitados por la captura de los datos. (26)

1 JDBC: es una interfaz de programación de aplicaciones (API) que permite la ejecución de operaciones sobre BD desde el lenguaje de programación Java, independientemente del sistema operativo o de la BD a la cual se accede, utilizando el dialecto SQL del modelo de BD que se utilice. (62)

Los cambios más significativos son resumidos en la tabla 1.1.

SymmetricDS 1.7.3	SymmetricDS 2.4
Los *triggers* incluyen la captura de los cambios en los datos y la definición de la ruta de los mismos. Por lo que al aumentar la cantidad de nodos se producen problemas de rendimiento, pues el número de filas crece linealmente con el aumento de los nodos clientes.	Solo captura los cambios en los datos. Incluye un nuevo mecanismo enrutador que separa la captura de los cambios en los datos del enrutamiento de los mismos. De esta forma han sido eliminadas las inserciones de filas con las especificaciones de los nodos.
Se crean varios disparadores por tablas, surgiendo problemas con algunas BD que solo permiten un *trigger* por tabla.	Se crea un *trigger* por tabla. Lo que permite direccionar los datos en múltiples direcciones.
Con el enrutamiento en el mismo hilo de los *triggers*, a los datos solo se le puede asignar una única ruta.	Con el enrutamiento separado de los *triggers* se le puede asignar varias rutas a un mismo dato.
Las tablas que contiene de historial de entrada y salida son difíciles de consultar. .	Han sido eliminadas las tablas con el historial y solo guarda el último intento de sincronización.

Tabla 1.1 SymmetricDS 1.7.3 vs SymmetricDS 2.4

SymmetricDS 2.4 incluye la capacidad de ejecutar complejas transformaciones en los datos cuando la sincronización de los datos está siendo cargada en la BD destino. Las transformaciones en los datos pueden ser usadas para mezclar datos del origen, hacer copias de los datos del origen en múltiples tablas del destino o ponerlos por defecto en las tablas del destino. Estas transformaciones a través de las extensiones del SymmetricDS 2.4 pueden ser adaptadas a las necesidades propias de cada usuario. (27)

Las nuevas características que contiene SymmetricDS 2.4 hacen que el rendimiento de las aplicaciones que se sincronizan con múltiples nodos no disminuya con el aumento del número de estos. Debido a que durante la sincronización los nodos clientes pasan menos tiempo conectados al servidor central.

A partir del análisis realizado se define para el desarrollo de la versión 2.0 del módulo Réplica la utilización de la versión 2.4 del SymmetricDS. Esta versión adiciona nuevas

funcionalidades e incluye las de la versión 1.7.3, convirtiéndolo en una herramienta superior funcionalmente.

1.7 Metodologías de desarrollo

1.7.1 Extreme Programming

Extreme Programming (XP) es una metodología ágil que busca simplificar el desarrollo de un software y que se logre reducir el costo, basada en la retroalimentación continua entre el cliente y el equipo de trabajo. Deriva varios principios básicos como la retroalimentación rápida, asumir la simplicidad, el cambio incremental, adherirse al cambio y trabajo de alta calidad. Intenta reducir la complejidad del software por medio de un trabajo orientado directamente al objetivo, basado en las relaciones interpersonales y la velocidad de reacción. Pretende minimizar el riesgo de fallo del proceso por medio de la disposición permanente de un representante del cliente a disposición del equipo de desarrollo.

Es un sistema de prácticas mínimas, las que se suponen utilizaren el principio de un proyecto, adaptarlas y agregar las que se vayan necesitando. Presenta un diseño evolutivo, provocando que se le dé poca importancia al análisis como fase independiente, ya que se trabaja en función de las necesidades del momento. Tiene la debilidad cuando se utiliza en dominios de aplicaciones complejas, por lo que es usada para proyectos a corto plazo y equipos pequeños. Se rediseña todo el tiempo, dejando el código siempre en el estado más simple posible. Se harán pruebas todo el tiempo, no sólo de cada nueva clase sino que también los clientes comprobarán que el proyecto va satisfaciendo los requisitos. (28)

1.7.2 Microsoft Solutions Framework

Microsoft Solutions Framework (MSF) es una metodología de desarrollo de software tradicional. Es una guía para administrar el equipo de trabajo y los procesos en el desarrollo de software. Se compone de varios modelos encargados de planificar las diferentes partes implicadas en el desarrollo de un proyecto: Modelo de Arquitectura del Proyecto, Modelo de Equipo, Modelo de Proceso, Modelo de Gestión del Riesgo, Modelo de Diseño de Proceso y finalmente el Modelo de Aplicación. (29)

MSF reúne las mejores prácticas en cuanto a la administración de proyectos se refiere. Más que una metodología rígida de administración de proyectos, MSF es una serie de modelos que puede adaptarse a cualquier proyecto de tecnología de información. Esta separa los proyectos en cinco fases. (30)

Esta metodología es adaptable a cualquier tipo de proyecto tanto a largo como a corto plazo. Sin embargo no es la más adecuada a seguir durante el desarrollo del módulo, pues está orientada a la administración del proyecto.

1.7.3 Proceso Unificado de Rational

El Proceso Unificado de Rational (RUP) se basa en las mejores prácticas que se han intentado y probado. Agrupa varias técnicas de desarrollo que proporcionan una guía para ordenar las actividades del equipo de desarrollo. Dirige las tareas de cada miembro del equipo por separado y del equipo como un todo, concreta los artefactos que deben desarrollarse y ofrece criterios para el control y la medición de los productos y actividades del proyecto. (31)

Sus creadores plantean que el ciclo de vida de RUP se caracteriza por: ser dirigido por casos de uso que orientan el proyecto a la importancia para el usuario y lo que este quiere; centrado en la arquitectura relacionando la toma de decisiones que indican cómo tiene que ser construido el sistema y en qué orden; iterativo e incremental pues divide el proyecto en pequeños proyectos donde los casos de uso y la arquitectura cumplen sus objetivos de manera más perfeccionada. El Proceso Unificado es una propuesta de proceso para el desarrollo de software orientado a objeto que utiliza el Lenguaje Unificado de Modelado (UML, *Unified Modeling Language*) para describir todo el proceso. (32)

RUP realiza un levantamiento exhaustivo de requerimientos, detecta errores en las fases iniciales, reduce el número de cambios, realiza el análisis y diseño tan completo como sea posible. Además, intenta anticiparse a futuras necesidades, los requerimientos de los clientes son fáciles de entender y existe un contrato prefijado con los mismos.

Del estudio realizado sobre algunas de las metodologías de desarrollo existentes, se define como metodología a seguir RUP, ya que es ideal para el desarrollo de proyectos grandes y persistentes, no siendo así XP pues su utilización trae consigo un cambio constante en los requerimientos del proyecto. Además, RUP permite que las decisiones sean tomadas desde el principio y se cumpla con la planificación que tiene como resultado un producto con las funcionalidades deseadas. Permite la generación de una serie de artefactos que sustentan el proceso de construcción del sistema. Posibilita la obtención de buenos resultados mediante la interacción entre ambas partes a través de reuniones, no siendo así XP que requiere que el cliente se encuentre dentro del equipo de desarrollo. Por otra parte, MSF está orientada a la administración de proyectos, de la cual no se tiene dominio sobre su utilización, lo que perjudicaría el desarrollo de la investigación en cuanto a tiempo y esfuerzo por parte del equipo de desarrollo.

1.8 Lenguajes de Modelado

1.8.1 Lenguaje de modelado BPMN

La Notación de Modelado de Procesos de Negocio (BPMN) es un estándar para el modelado de procesos de negocio y proporciona una notación gráfica para especificar los procesos de negocio en un Diagrama de Procesos de Negocio. El objetivo de BPMN es apoyar los procesos de negocio para la gestión técnica de usuarios y usuarios de negocios, proporcionando una notación que sea intuitiva para usuarios empresariales. (33)

BPMN facilita una notación estándar que sea comprensible por todas las partes interesadas. Estas partes son las empresas, los analistas de negocios que crean y refinan los procesos, los desarrolladores de la técnica responsable de la ejecución de los procesos y los directivos de las empresas que supervisan y gestionan los procesos.

Este lenguaje tiene como desventajas la existencia de un alto grado de ambigüedad en el proceso de negocio modelado. No se utilizan todas las potencialidades de modelamiento necesarias para disminuir el grado de ambigüedad. El cliente no garantiza el cumplimiento de los estándares mínimos de modelado de proceso.

1.8.2 Lenguaje Unificado de Modelado

El Lenguaje Unificado de Modelado (UML) es un lenguaje de notación esquemática con el que se construyen sistemas por medio de conceptos orientados a objetos. Es un lenguaje para visualizar, especificar y documentar los artefactos que forman parte del proceso de desarrollo del software. Está compuesto por la combinación de varios elementos gráficos que forman diagramas. Este lenguaje permite la modelación de sistemas con tecnología orientada a objetos, donde se describe lo que hará un sistema pero no dice cómo implementarlo. (34)

Es seleccionado el lenguaje UML pues permite especificar las características que debe tener el sistema, permitiendo a partir de los modelos diseñados construir y documentar a través de los elementos gráficos el mismo. También, modelar los procesos de negocio y funciones. Escribir clases en un lenguaje determinado, esquemas de BD y componentes de software reusables. Además, es utilizado por la metodología RUP para la descripción de los procesos. UML está orientado a la construcción de sistemas orientados a objetos, mientras que BPMN toma un perfil orientado a procesos en el modelado de sistemas. UML se enfoca al diseño de software y BPMN tiene un enfoque hacia los procesos de negocio.

1.9 Herramientas para la modelación del software

1.9.1 ArgoUML

ArgoUML es un proyecto abierto, la disponibilidad del código fuente asegura que una nueva generación de diseñadores y de investigadores de software tenga un marco probado del cual puedan conducir el desarrollo. Contiene funciones avanzadas en las etapas de diseño y modelación de software. Dentro de sus principales características se encuentra que tiene muy buenas ayudas, soporta todas las especificaciones UML y está integrado con la Web. Además, es multiplataforma y genera código en distintos lenguajes. Los diseños son exportables a XML y pueden ser importados por algunos *frameworks*, su instalación es costosa, es poco amigable y difícil de aprender. (35)

Esta herramienta tiene como desventajas que es pesada, utiliza UML 1.4, su interfaz no es intuitiva, no contiene íconos que permitan conocer los estereotipos, no soporta la ingeniería inversa y requiere de capacitación para su utilización.

1.9.2 Visual Paradigm for UML

Visual Paradigm compendia un conjunto de funcionalidades para el desarrollo de software. Permite el modelado de los requerimientos, procesos de negocio y modelado de BD. Es propiedad de la compañía Visual Paradigm con licencia gratuita y comercial. Es una aplicación multiplataforma, que proporciona la generación automática de documentos y código a partir de los diagramas en diferentes formatos. Permite esbozar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Soporta la notación UML 2.1, ingeniería inversa, generación de código, exportación/importación XML, generador de informes, editor de figuras, integración con MS Visio, integración IDE con Visual Studio, IntelliJ IDEA, Eclipse, NetBeans y otros, además incluye el modelado colaborativo con CVS y Subversion. (36)

La herramienta ofrece el diseño centrado en casos de uso, el uso de un lenguaje estándar común a cualquier equipo de desarrollo facilitando la comunicación. Puede lograrse la sincronización entre el modelo y el código durante el ciclo de desarrollo.

A partir del análisis realizado sobre estas herramientas de modelado se define que la más adecuada para el modelado del sistema es Visual Paradigm. La misma permite modelar diagramas de clases, realizar código inverso, generar código desde diagramas y generar documentación. Utiliza UML 2.1 a diferencia de ArgoUML que emplea 1.4. Puede ser ejecutado tanto en Windows como en Linux y soportar el ciclo de vida completo de un software.

1.10 Lenguaje de programación y herramientas para el desarrollo del sistema

El módulo Réplica fue desarrollado integrado a la Plataforma Educativa Zera, resultando costoso en tiempo de implementación cambiar los lenguajes y herramientas utilizadas. Durante el diseño de la versión 2.0 del módulo Réplica como parte de la Plataforma Educativa Zera, deben tomarse en cuenta las tecnologías y herramientas definidas en la arquitectura de la misma, actualizando el versionado de estas.

1.10.1 PHP

El lenguaje PHP (Pre-procesador de hipertexto) es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor. Diseñado especialmente para el desarrollo de aplicaciones web. Publicado bajo la licencia PHP, la Fundación de Software Libre considera esta licencia como software libre. La meta del lenguaje es permitir rápidamente a los desarrolladores la generación dinámica de páginas. Cuenta con capacidad de conexión con la mayoría de los gestores de BD utilizados actualmente y de expandir su potencial utilizando módulos y manejo de excepciones. Además, tiene soporte para una gran cantidad de BD y mantiene un bajo consumo de recursos de máquina. (37)

1.10.2 Symfony

Symfony es un *framework* para el desarrollo de las aplicaciones web con PHP. Define una serie de herramientas y ventajas que simplifican el desarrollo de estas. Separa la lógica del negocio, la lógica del servidor y la presentación de la aplicación web. También automatiza las tareas más frecuentes, permitiendo al programador centrarse en los aspectos específicos de la aplicación. Es independiente del sistema gestor de BD. Emplea auto generadores de código y herramientas que reducen el tiempo de desarrollo. Es fácil de instalar y configurar en la mayoría de las plataformas, sigue la mayoría de las mejores prácticas y patrones de diseño para la web. Además, es lo suficientemente estable como para desarrollar aplicaciones a largo plazo. (38)

1.10.3 PostgreSQL

PostgreSQL es un gestor de BD objeto-relacional de código abierto diseñado para administrar grandes volúmenes de datos. Incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. Soporta distintos tipos de datos y permite la creación de tipos de datos propios. Permite la declaración de funciones propias, así como la definición de disparadores. Soporta el uso de índices, reglas y vistas e incluye herencia entre tablas.

También, posibilita la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos. Además, sirve de soporte a los lenguajes más populares como PHP. (39)

1.10.4 XHTML

XHTML, acrónimo en inglés de *eXtensible Hypertext Markup Language* (Lenguaje Extensible de Marcado de Hipertexto), es el lenguaje de marcado pensado para sustituir Al Lenguaje de Marcado de Hipertexto (HTML) como estándar para las páginas web. Es una versión avanzada de HTML y basada en el Lenguaje eXtensible de Marcado (XML). Es la versión XML de HTML por lo que tiene básicamente las mismas funcionalidades, pero cumple las especificaciones más estrictas de XML. Entre las principales ventajas del XHTML sobre el HTML se encuentra la incorporación de elementos de distintos espacios de nombres XML, el navegador no necesita implementar heurísticas para detectar qué quiso poner el autor. XHTML extiende HTML 4.0 combinando la sintaxis de HTML (diseñado para mostrar datos), con la de XML (diseñado para describir los datos) (40)

1.10.5 Hojas de Estilos en Cascada (CCS)

CCS (del inglés *Cascading Style Sheets*) es un lenguaje de hojas de estilo creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. Describe cómo se va a mostrar un documento en la pantalla o cómo se va a imprimir o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre el estilo y formato de sus documentos. Es utilizado para dar estilo a documentos HTML y XML, separando el contenido de la presentación. Los estilos definen la forma de mostrar los elementos HTML y XML. Permite a los desarrolladores web controlar el estilo y el formato de múltiples páginas web al mismo tiempo. Cualquier cambio en el estilo marcado para un elemento en la CSS afectará a todas las páginas vinculadas a esa CSS en las que aparezca ese elemento. (41)

1.10.6 Javascript

Javascript es un lenguaje de programación que no requiere de compilación, pues funciona del lado del cliente, los navegadores son los encargados de interpretar estos códigos. Nació con la necesidad de permitir a los autores de sitio web crear páginas que permitan intercambiar con los usuarios, pues se necesitaba crear webs más completas. El HTML solo permitía crear páginas estáticas donde se podía mostrar textos con estilos, pero era necesario interactuar con los usuarios. Para evitar incompatibilidades el *World Wide Web Consortium* (W3C) diseñó un estándar denominado DOM (en inglés *Document Object*

Model, en su traducción al español Modelo de Objetos del Documento). Permite a los desarrolladores crear acciones en las páginas web. Su sintaxis es similar a la usada en Java y C al ser un lenguaje del lado del cliente este es interpretado por el navegador, no se necesita tener instalado ningún Framework. Javascript tiene la ventaja de ser incorporado en cualquier página web, puede ser ejecutado sin la necesidad de instalar otro programa para ser visualizado. Es soportado por la mayoría de los navegadores como Internet Explorer, Netscape, Opera, Mozilla Firefox, entre otros. (42)

1.10.7 Ajax

Con el surgimiento de lenguajes como PHP del lado del servidor y Javascript del lado del cliente, surgió Ajax, acrónimo de *Asynchronous JavaScript And XML (*JavaScript asíncrono y XML*)*, una técnica de desarrollo web para crear aplicaciones interactivas. Los datos adicionales se requieren al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página. En JavaScript normalmente se efectúan las funciones de llamada de Ajax mientras que el acceso a los datos se realiza mediante *XMLHttpRequest*[2], objeto disponible en los navegadores actuales. (43)

1.10.8 Doctrine ORM

Un ORM (en inglés Object Relational Mapper) es una técnica de programación que permite convertir datos entre el sistema de tipos de datos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una BD relacional, es decir, las tablas de la BD pasan a ser clases y los registros, objetos que se pueden manejar con facilidad. (44)

Doctrine es un sistema ORM para PHP 5.2.3 y posterior, que incorpora una DBL (capa de abstracción a BD). Uno de sus rasgos importantes es la habilidad de escribir opcionalmente las preguntas de la BD orientada a objeto. Esto les proporciona una alternativa poderosa a diseñadores de SQL manteniendo un máximo de flexibilidad sin requerir la duplicación del código innecesario. Exporta una BD existente a sus clases correspondientes. Convierte clases (convenientemente creadas siguiendo las pautas del ORM) a tablas de una BD. (45)

1.11 Selección de las tecnologías, metodologías y herramientas a utilizar

La selección de las tecnologías, la metodología y las herramientas fue realizada basándose en sus características, las ventajas y desventajas de su utilización. Además, las

[2]XMLHttpRequest:(*Extensible Markup Language / Hypertext Transfer Protocol*), es una interfaz empleada para realizar peticiones HTTP y HTTPS a servidores Web.

definidas en la arquitectura de la Plataforma Educativa Zera, su reconocimiento a nivel mundial, que fueran de software libre y de mayor dominio por el equipo de desarrollo del trabajo. Fueron seleccionadas:

- La metodología a seguir durante todo el proceso de desarrollo RUP.
- Como lenguaje de modelado UML 2.0.
- Visual Paradigm for UML 8.0 como herramienta para el modelado del software.
- Las tecnologías y herramientas definidas en la arquitectura de la Plataforma Educativa Zera: Symfony 1.4.*, PHP 5.3, XHTML, CCS, Javascript, Ajax y Doctrine ORM.

1.12 Conclusiones parciales

La consulta bibliográfica posibilitó definir los conceptos y tendencias que sustentan el marco teórico del objeto de estudio analizado.

La metodología de investigación empleada permitió definir las tecnologías, herramientas y metodologías para darle cumplimiento al objetivo general de la investigación.

Los métodos de observación y el histórico lógico permitieron hacer un análisis de la versión 1.0 del módulo y definir sus insuficiencias, así como la comparación entre las versiones 1.7.3 y 2.4 de la herramienta de replicación utilizada SymmetricDS y seleccionar como más idónea SymmetricDS 2.4 para la propuesta de análisis y diseño.

Capítulo II: Características del sistema

Introducción

En el presente capítulo se describen las especificaciones que se deben seguir durante el diseño del módulo, así como una guía de cómo debe realizarse la replicación en situaciones concretas. Se exponen las características del módulo que le darán solución al problema existente, describiéndose los procesos que lo generan y los que deberán automatizarse. Se detalla el Modelo de Dominio, los requisitos funcionales y no funcionales, así como el Diagrama de Casos de Uso del Sistema y la descripción todos los casos de uso que componen este diagrama.

2.1 Descripción de la propuesta de solución

La herramienta de replicación SymmetricDS 2.4 se configura a través de archivos en los cuales son introducidos los datos de configuración. El módulo Réplica es una interfaz entre esta herramienta y los usuarios de la Plataforma Educativa Zeraque facilita y realiza de forma transparente la replicación de los datos de la Plataforma Educativa Zera. Además, posibilita a los usuarios administrar las replicaciones aunque no tengan conocimientos avanzados de replicación.

En la Plataforma Educativa Zera el administrador es el responsable de realizar las operaciones de configuración y administración de las replicaciones en los servidores de BD. Existe un Administrador local que es el responsable de configurar la réplica en el servidor de BD local y monitorea el estado en que se encuentran las sincronizaciones de los datos de la escuela. También existe un Administrador central que configura, administra y monitorea las replicaciones a nivel central. Además, el director y los profesores de una escuela que pueden consultar los estados en los cuales se encuentran las réplicas y los informes de las últimas sincronizaciones, operación esta que también es realizada por ambos administradores.

El objetivo principal del módulo es permitir a los administradores la adecuada configuración de la réplica. Se propone incluir nuevas funcionalidades y mejoras a las existentes, que le darán solución a los problemas actuales del módulo. Estas nuevas funcionalidades y mejoras del módulo son propuestas a continuación:

Con el objetivo de solucionar el conflicto producido cuando son realizados en el servidor central cambios estructurales en la BD, se propone la sincronización de los cambios realizados en el esquema de la BD central con los esquemas de las BD locales, para que exista uniformidad en los esquemas en todo el SBDD. Para ejecutar este proceso se

permitirá la selección de las escuelas con las cuáles se van a realizar las sincronizaciones de los esquemas. Al terminar el proceso se informará cuáles nodos completaron la sincronización y cuáles no. En los casos que no se complete el proceso de sincronización se mostrará una descripción del motivo por el cual no se completó el proceso.

Para solucionar el problema producido al replicar las tablas sin tener en cuenta sus precedencias, se propone establecer niveles de replicación entre las tablas. Primeramente se deberá replicar las tablas que no tienen precedencias o relaciones con otras. Luego, se procederá a replicar las tablas que contengan llaves foráneas y por último las tablas que tengan como llave primaria llaves foráneas.

Si se perdiera la conexión entre el servidor central y algún servidor local por un tiempo considerable, los cambios en los datos serían capturados y almacenados en cada uno de los servidores. Al establecerse la conexión entre los servidores e intentar la sincronización de los datos, en el caso de que surgiera un conflicto de orden, se compararían los tiempos en los cuales fueron actualizados o eliminados los datos y se tomaría como correcta la acción más reciente efectuada sobre el dato.

Para que el usuario tenga conocimientos de cuáles datos han sido actualizados, insertados o eliminados, se le dará la posibilidad de consultar los datos que han sido sincronizados. Se le mostrará el nombre de la tabla, el dato anterior y el nuevo, especificando además qué acción de actualización, inserción o eliminación fue realizada sobre este.

A través del módulo se podrá activar o desactivar la réplica cuando el usuario así lo desee, tanto desde o en el servidor central como en los servidores locales.

Para mejorar la consulta del estado de replicación, cuando el usuario seleccione esta opción se mostrará el último intento de sincronización realizado por la escuela, mostrando la fecha, la hora y si se completó o fue fallida. Se podrá revisar los estados de las escuelas desde el servidor central, pudiéndose mostrar los detalles de las últimas sincronizaciones de las escuelas o de una en particular.

En función de perfeccionar la generación de los reportes de replicación, se le permitirá al usuario la selección detallada de los datos que desea que se muestren en el informe.

Para facilitar la consulta de las últimas sincronizaciones, se mostrará únicamente la última sincronización efectuada por una escuela.

2.1.1 Descripción del funcionamiento del SymmetricDS 2.4

Para un correcto planeamiento de la implementación del SymmetricDS 2.4 es necesaria la identificación de los nodos, que en este caso son cada uno de los servidores de BD de la

Plataforma Educativa Zera donde se ejecuta una instancia del SymmetricDS, que es responsable de la administración de la réplica en dicho nodo. Estos nodos son organizados basándose en los datos que necesitan ser sincronizados, teniendo en cuenta qué datos son comunes para dos o más nodos, esta organización es comprendida como un grupo de nodos. Los grupos de nodos deben ser creados para cada juego de nodos que sincronizan tablas comunes en modos similares. Los enlaces son la vinculación entre los nodos fuente a los destinos, especificando las acciones a realizar entre ambos. En la Plataforma Educativa Zera cuando existe un cambio en los datos en un nodo local este utiliza el método de empujar, se conecta al nodo central y pone los cambios. En el caso que los cambios sean realizados en el nodo central, el local se conecta a este y captura los cambios.

Las tablas a sincronizar son agrupadas en canales de datos que permiten establecer prioridades de sincronización a los datos. El comportamiento de las sincronizaciones puede ser controlado a nivel de canales. Estos proporcionan un orden al proceso de sincronización, establecen un límite en la cantidad de datos que son cargados. Es muy importante tener en cuenta al definir un canal que todas las tablas relacionadas por llaves foráneas sean incluidas en el mismo canal. (27)

Los disparadores se utilizan para capturar los cambios en los registros y sincronizarlos a otros nodos, los cuales son definidos en la tabla TRIGGER, donde cada disparador que es definido está asociado a una tabla. Cada disparador puede especificar si se instala para las actualizaciones, las inserciones y/o eliminaciones, también sobre qué condiciones insertar, actualizar y/o eliminar registros.

El canal que va a ser usado se define en la tabla ROUTER, lo que unido al mapeo entre los disparadores y las rutas en la tabla TRIGGER_ROUTER definen el proceso para determinar cuáles nodos recibirán los cambios en los datos.

El SymmetricDS 2.4 cuenta para guardar los cambios y las rutas con una BD de disparadores creada automáticamente para capturar los cambios en los datos registrándolos en la tabla DATA. La asignación de la ruta de los datos, organiza y guarda la decisión de hacia dónde se enviarán los datos y es organizado en unas tablas llamadas DATA_EVENT y OUTGOING_BATCH. (27)

2.1.2 Descripción de la configuración del SymmetricDS 2.4

La primera operación que debe realizarse luego de instalar el SymmetricDS 2.4 es editar los archivos de las propiedades para el nodo central y para los nodos clientes. Siendo los archivos los que se muestran a continuación:

Samples/root.properties: es modificado únicamente en el nodo central, identificado como nodo raíz.

Samples/client.properties: es modificado en cada uno de los nodos clientes asociados al central.

En ambos archivos se especifican los siguientes parámetros que son los más importantes en el proceso de configuración de un nodo:

db_driver: nombre de la clase del controlador JDBC en este caso para PostgreSQL.

db_url: URL JDBC que utiliza para conectarse a la BD.

db.user: usuario de la BD con privilegios para acceder, crear y actualizar las tablas del SymmetricDS.

Password: contraseña para el usuario de la BD.

Para los nodos clientes se debe especificar la dirección del nodo central al que se va a conectar, a través del parámetro **registration.url**.

En el tiempo de ejecución la configuración es usada para capturar los cambios y dirigirlos a los nodos. Los cambios en los datos se colocan juntos en un solo lugar llamado lote el cual puede ser cargado por otro nodo. Se envían los lotes salientes y se reconocen, así como se reciben los lotes entrantes y se cargan. El historial de los cambios de estado de lote y estadística es guardado.

2.1.3 Resolución de conflictos

SymmetricDS 2.4 no cuenta por sí solo con mecanismos para la resolución de conflictos, sin embargo, contiene varios puntos de extensión donde el programador puede implementar y agregarle sus propios métodos para la resolución de los mismos. En la interfaz *IExtensionPoint* son extendidos estos puntos.

En la interfaz *IExtensionPoint* son se designa si el punto de extensión se registra automáticamente o no. Estos puntos pueden ser designados a un grupo de nodos a través de la interfaz *INodeGroupExtensionPoint* la cual puede ser implementada opcionalmente. Las estrategias de resolución de conflictos pueden ser implementadas utilizando el punto de extensión *IDataLoaderFilter* que tiene acceso a los datos viejos y nuevos. La implementación de los métodos de resolución de conflictos a través de *IDataLoaderFilter*es realizada mediante la recepción de los *callbacks* (devoluciones de llamadas) cuando los datos son insertados, actualizados o borrados. Los datos pueden ser filtrados tanto cuando se están cargando en el destino como cuando se extraen del origen. También puede

especificar por el valor que retorna la función si la carga de los datos debe continuar y debe cargar los datos si devuelve verdadero o ignorarlos si retorna falso. Es empleado el lenguaje de programación Java, un ejemplo de estas implementaciones es anexado al presente documento. (27)

Deben implementarse métodos dentro de dicha clase para darle solución a los posibles conflictos que se puedan producir en la replicación. También, debe ser especificado por el programador un orden de replicación para las tablas, estableciendo niveles entre ellas según sus precedencias. Además, definir que ante un conflicto de actualización será considerada correcta la más reciente de las actualizaciones.

2.1.4 Sincronización de cambios estructurales en la BD

Cuando sean realizados cambios en la estructura o esquema de la BD es necesario que estos cambios sean actualizados en cada uno de los nodos clientes. Este proceso de replicación de la modificación de la estructura de la BD para que concuerde con la estructura de las demás BD en los clientes, es un proceso completamente separado de la sincronización de los datos. Por lo cual, dicho proceso es independiente de la herramienta de replicación y debe ser realizado por el programador. Estos cambios deben replicarse en todo el sistema de forma organizada. Primeramente debe realizarse una salva o copia de seguridad de las BD antes de que los cambios sean aplicados.

Para iniciar el proceso de replicación de los cambios se debe identificar cuál es la BD origen y cuáles son las BD destinos. Se realizará una correspondencia entre cada uno de los elementos, quedando especificado qué cambios del origen serán replicados al destino. Luego deben quedar identificadas las modificaciones realizadas a los elementos y las acciones ejecutadas en el origen durante todo el proceso de cambios estructurales en la BD.

Una vez identificados todos estos elementos se procede con la comparación de los esquemas, donde deben especificarse por cada elemento del origen qué acción de actualización, inserción o eliminación debe ser realizada para sincronizar dicho elemento en cada BD destino. Para realizar la comparación deben establecerse criterios de comparación entre los objetos de la BD. (46)

Durante el proceso de comparación deben identificarse qué objetos han sido eliminados, cuáles han sido insertados y cuáles continúan, que pueden haber sido o no modificados. Después se analizarán los que continúan con el fin de encontrar qué modificaciones le fueron realizadas. Por último se realiza un análisis de las relaciones entre los objetos de la BD.

Una vez finalizada la comparación de los esquemas entre las BD, se debe proceder a la actualización de los destinos, donde siempre lo primero será realizar una copia de seguridad del destino. Luego se ejecutará la actualización de la estructura del destino, realizando cada una de las acciones que ya fueron identificadas en la comparación de los esquemas.

2.2 Modelo de Dominio

Partiendo del análisis realizado sobre el funcionamiento y cómo son realizados los procesos de réplica y sincronización de los datos en la Plataforma Educativa Zera, así como el estudio del SymmetricDS 2.4. Es definido que el negocio es similar al de la versión 1.0, presentando un bajo nivel de estructuración y difícil de comprender, por lo que se especifican estos procesos a través de un Modelo de Dominio. Este modelo describe los conceptos utilizados por los usuarios que deben ser tomados en cuenta por la aplicación y tiene como propósito ayudar a la comprensión del negocio. En el Modelo de Dominio (Ver Figura 2.1) se modelan los conceptos asociados al dominio.

Clases conceptuales del dominio:

nodo_Central: representa al servidor central de la Plataforma Educativa Zera o la institución central.
nodo_Escuela: representa a un servidor local de la Plataforma Educativa Zera o una escuela asociada a la institución central.
Sistema: es la asociación del gestor de BD PostgreSQL y la herramienta SymmetricDS con la información gestionada en la Plataforma Educativa Zera.
Usuario: representa a las personas que interactúan con el sistema (Administradores, Profesores y Directores).
Administrador: representa a las personas con las facultades para realizar operaciones de administración en la Plataforma Educativa Zera. Este puede ser local o central.
Base de datos: representa a las BD de la Plataforma Educativa Zera presentes en cada uno de los nodos.
BD_Origen: representa a las BD que funcionarán como origen de la actualización de los datos.
BD_Destino: representa a las BD que funcionarán como destino de la actualización de los datos.

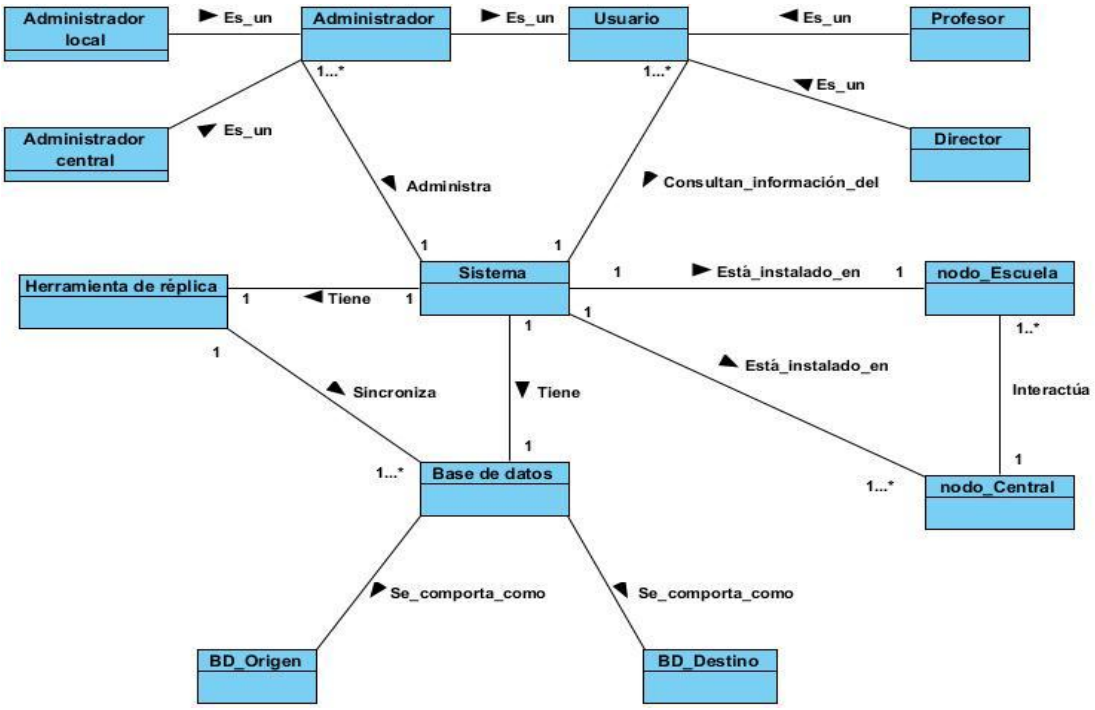


Figura 2.1 Diagrama del Modelo de Dominio.

2.3 Especificación de requisitos

2.3.1 Requisitos funcionales

El profesor Ian Sommerville define que: *¨Los requisitos funcionales de un sistema describen lo que el sistema debe hacer.¨* (47)

Los requisitos funcionales del módulo Réplica de la Plataforma Educativa se especifican a continuación:

RF-1 Configurar ambiente de replicación: debe permitir la configuración inicial de la replicación para el servidor central y para cada uno de los servidores locales, posibilitando al usurario introducir los valores para realizar la configuración (Usuario, Contraseña y Nombre de la BD).

RF-2 Activar la réplica: debe permitir a los administradores activar la replicación, inicializando los procesos de réplica en el servidor central y en cada uno de los servidores locales.

RF-2.1 debe permitir la activación de la réplica en las escuelas desde el servidor central.

RF-2.2 debe permitir la activación de la réplica en cada escuela desde su servidor local.

RF-3 Desactivar la réplica: debe permitir a los administradores desactivar la réplica desde el servidor central a cada uno de los servidores locales.

RF-3.1 debe permitir la desactivación de la réplica de los servidores central y locales desde el servidor central.

RF-3.2 debe permitir la desactivación de la réplica en cada escuela desde su servidor local.

RF-4 Mostrar estado de replicación: debe permitir que los usuarios consulten el estado de la réplica de su escuela, mostrando un informe con el nombre de la escuela, la fecha y hora de los últimos intentos de sincronización, identificándolas como completada o fallida.

RF- 4.1 debe mostrar el estado de replicación de las escuelas desde el servidor central.

RF-4.2 debe mostrar el estado de replicación en cada escuela desde su servidor local.

RF-5 Generar reporte de sincronización: debe permitir la generación de reportes sobre las sincronizaciones, a partir del filtrado por estado (completada o fallida) y el nombre de la escuela. Además debe posibilitar al usuario seleccionar qué datos desea que se muestren en el reporte (Fecha, Hora, Estado, Descripción del error, Datos sincronizados)

RF-6 Mostrar escuelas configuradas: debe mostrar un listado con el nombre de las escuelas que han sido configuradas para la réplica.

RF-7 Mostrar escuelas sin configurar: debe mostrar un listado con el nombre de las escuelas que no han sido configuradas para la réplica.

RF-8 Mostrar información general: debe mostrar un listado con el total de escuelas configuradas, de escuelas no configuradas, de sincronizaciones y la cantidad de sincronizaciones fallidas.

RF-9 Sincronizar cambios estructurales en la BD: debe permitir al usuario luego de realizar cambios en la estructura de la BD sincronizarlos con los nodos locales.

RF-9.1: debe permitir que el usuario seleccione los nodos a sincronizar.

RF-9.2: debe generar un reporte con los nombres de los nodos sincronizados satisfactoriamente y los que no, especificando los motivos de los fallos.

RF-10 Mostrar últimas escuelas sincronizadas: debe mostrar una lista con los nombres de las escuelas que han sincronizado sus datos con el servidor central en el día.

RF-11 Mostrar últimas sincronizaciones de una escuela: debe mostrar para una escuela el listado con la hora y el estado de las sincronizaciones realizadas en el día.

RF-12 Mostrar los datos sincronizados: debe mostrar de los datos sincronizados, sus modificaciones, cambios y acciones realizadas sobre ellos.

RF-12.1: debe mostrar los datos sincronizados de las escuelas desde el servidor central.

RF-12.2: debe mostrar los datos sincronizados de una escuela desde el servidor local que le corresponde.

RF-13 Solucionar conflictos de orden: debe solucionar los conflictos de orden, asumiendo como correcta la más reciente de las acciones realizadas sobre los datos.

RF-14 Establecer orden de replicación entre las tablas de la BD: debe solucionar los conflictos de replicación de tablas en orden incorrecto, estableciendo niveles de replicación entre las tablas según las relaciones que existan entre estas.

RF-14.1: en el primer nivel de replicación debe ubicar las tablas que no tengan precedencias.

RF-14.2: en el segundo nivel de replicación debe ubicar las tablas que contengan llaves foráneas.

RF-14.3: en el tercer nivel de replicación debe ubicar las tablas que sus llaves primarias sean llaves foráneas.

2.3.2 Requisitos no funcionales

Ian Somerville define los requisitos no funcionales como: *¨aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a propiedades emergentes de éste…¨*

Los requisitos no funcionales del módulo Réplica de la Plataforma Educativa Zera se especifican a continuación:

Usabilidad: el módulo solo podrá ser utilizado por los administradores, los profesores y el director de la institución al que pertenece el mismo.

Portabilidad: el módulo debe funcionar sobre cualquier sistema operativo.

Seguridad: solamente pueden realizar operaciones a través del módulo los usuarios que tengan los permisos para configurar y/o monitorear la replicación.

Interfaz externa: en la interfaz principal debe mostrase una introducción a la replicación de datos en general y en cada una de las adyacentes orientarle al usuario que operaciones puede realizar en las mismas.

Restricciones en el Diseño: para el modelado del software debe utilizarse el lenguaje UML 2.0 y la herramienta Visual Paradigm for UML 8.0.

Restricciones de Implementación: se debe usar para el desarrollo, el lenguaje de programación PHP 5.3, el *framework* Symfony 1.4.*, como herramienta de réplica SymmetricDS 2.4 y como SGBD PostgreSQL 9.1.

Disponibilidad: la información que sea solicitada por el usuario debe estar disponible las 24 horas del día.

2.4 Definición y descripción de los Casos de Uso del sistema

2.4.1 Patrones de Casos de Uso

Los patrones de Casos de Uso (CU) se emplean para describir cómo deben ser estructurados y organizados los CU, los mismos capturan las mejores prácticas para modelar CU. La utilización de patrones permite aumentar la productividad, reutilizar elementos existentes y no invertir tiempo en resolver problemas ya resueltos.

Durante realización del Diagrama de CU se aplicaron los patrones que a continuación se explican.

Extensión Concreta: consiste en dos CU y una relación extendida entre ellos. Puede ser instalado en sí mismo, así como extendido en el CU base. Este patrón se aplica cuando un flujo puede extender el flujo de otro CU así como ser realizado en sí mismo. (48) Este patrón se aplica a los CU Mostrar datos sincronizados y Configurar ambiente de replicación, los cuales pueden ser realizados por si mismo o extendidos desde los CU Generar reportes de sincronización y Mostrara escuelas sin configurar.

Múltiples Actores: cuando varios actores jueguen un mismo rol sobre un CU, se representa este rol con otro actor del cual heredan los demás actores que juegan este mismo rol. Se aplica cuando solo existe una entidad externa interactuando con cada una de las instancias del CU. (48) Este patrón se utilizó para unir las acciones comunes del Director, Profesor, Administrador central y Administrador local como el Usuario.

2.4.2 Definición de actores

Administrador local: realiza las tareas de administración en la Plataforma Educativa Zera instalada en su escuela y puede ver el espacio de la escuela en la Plataforma Educativa Zera instalada en el servidor central. Es quien consulta el estado en que se encuentra la réplica de los datos en la escuela.

Administrador central: se encarga de la administración global del sitio y tiene permisos sobre todos los componentes de la Plataforma Educativa Zera. Es el encargado de configurar la réplica y todas las demás funciones referentes a la misma tanto en la escuela como en la plataforma central.

Director: mediante el rol Usuario, consulta los reportes de estado de la réplica y el registro de las últimas sincronizaciones.

Profesor: mediante el rol Usuario, consulta los reportes de estado de la réplica y el registro de las últimas sincronizaciones.

2.4.3 Definición de CU

Los CU son artefactos narrativos que describen, bajo la forma de acciones y reacciones, el comportamiento del sistema desde el punto de vista del usuario, establece un acuerdo entre clientes y desarrolladores sobre las condiciones y posibilidades (requisitos) que debe cumplir el sistema. (49)

2.4.3.1 Diagrama de CU

Un diagrama de CU del sistema representa gráficamente a los procesos y su interacción con los actores, describe la funcionalidad propuesta del nuevo sistema a desarrollar y los procesos a automatizar, representados a través de CU que responden a los requisitos funcionales del mismo. (49)

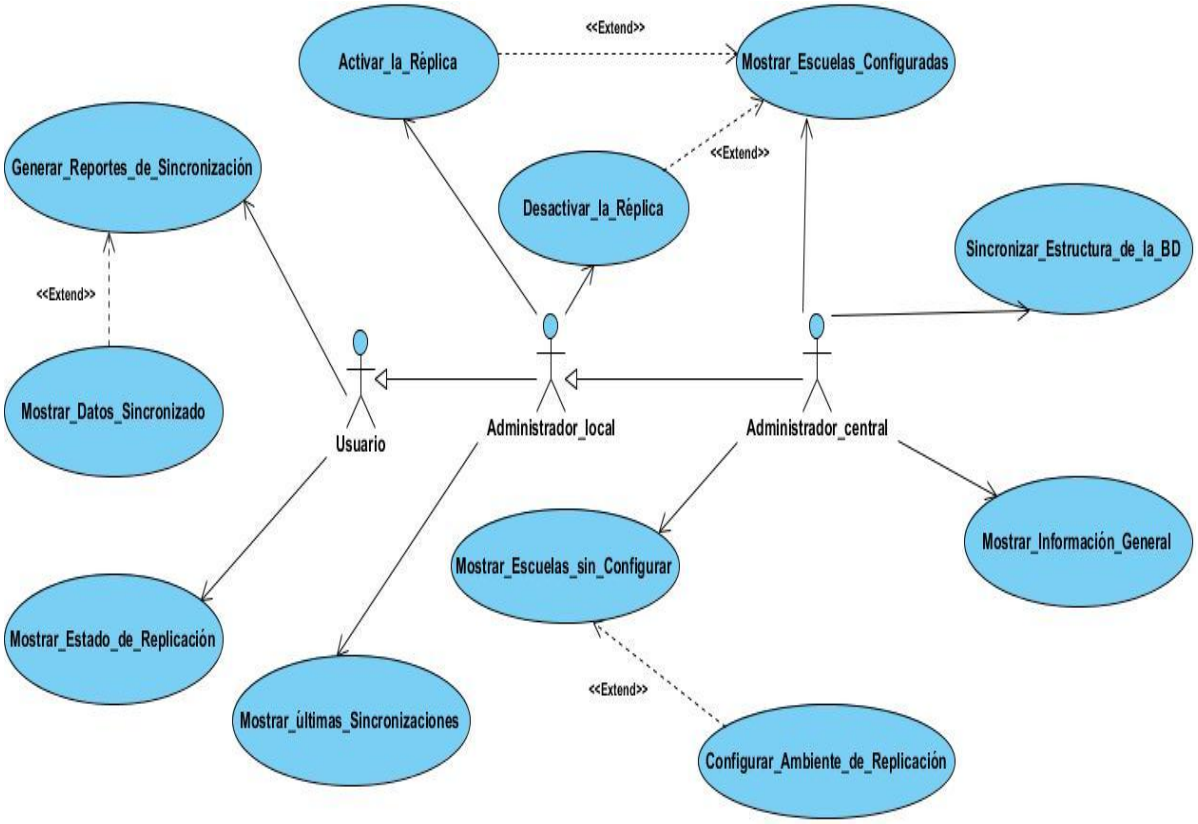


Figura 2.3 Diagrama de Casos de Uso

2.4.4 Descripción de los CU

Tabla 2.1 CU Sincronizar estructura de la BD

Objetivo	Sincronizar los cambios realizados en la estructura de la BD con las BD

	de los servidores locales.
Actores	Administrador central: sincroniza la estructura de la BD.
Resumen	El actor selecciona la opción de ¨Sincronizar Estructura de la BD¨, el usuario debe seleccionar una o varias BD destinos, luego de que este realiza la selección, el sistema realiza una copia de seguridad de cada uno de los destinos y en orden realiza la sincronización. Al terminar cada una de ellas, genera un informe.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	Debe haberse generado el escritorio de trabajo del usuario autenticado. El servidor central debe estar configurado. Para sincronizar la estructura de la BD, el actor debe ser el responsable temporal de la acción.
Poscondiciones	Fueron sincronizadas las BD locales.

Las descripciones totales de los CU anteriormente mencionados y los restantes CU se encuentran en los Anexos.

2.5 Conclusiones parciales

La selección de la metodología, las herramientas y lenguaje de modelado definida en el capítulo anterior posibilito la descripción de la propuesta de solución al problema planteado y los principales conceptos del dominio.

Se detalló el funcionamiento de la herramienta de replicación SymmetricDS 2.4 como parte de la solución al problema.

Del levantamiento de requerimientos realizado, quedaron definidas las funcionalidades del sistema y sus descripciones. A partir de estas es posible iniciar el análisis y diseño del módulo Réplica de la Plataforma Educativa Zera.

Capítulo III: Análisis y Diseño del módulo Réplica

Introducción

El presente capítulo expone la propuesta de análisis y diseño de la versión 2.0 del módulo Réplica. Incluye los principales artefactos del flujo de trabajo Análisis y Diseño, tales como las definiciones de las Clases del Análisis y las del Diseño, los distintos diagramas necesarios para representar el flujo principal y alterno de los CU de la aplicación.

3.1 Análisis del sistema

3.1.1 Diagramas de Clases del Análisis

Las clases del análisis están centradas en los requisitos funcionales y son más evidentes en el dominio del problema representando relaciones y conceptos del mismo. Estas se clasifican en: (31)

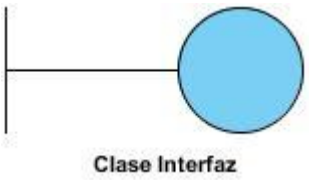
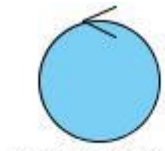
Prototipo	Clases	Descripción
 Clase Entidad	**Entidad**	Se utilizan para modelar información que posee larga vida y que es a menudo persistente.
 Clase Interfaz	**Interfaz**	Se utilizan para modelar la interacción entre el sistema y sus actores, modelan las partes del sistema que dependen de sus actores.
 Clase Controladora	**Control**	Representan coordinación, secuencia, transacciones y control de otros objetos.

Tabla 3.1 Clases del análisis

El Diagrama de Clases del Análisis (DCA) muestra las clases participantes y sus relaciones en la realización de un CU. (31)

A continuación se muestra un DCA asociado al problema en cuestión, los restantes diagramas se encuentran recogidos en los Anexos.

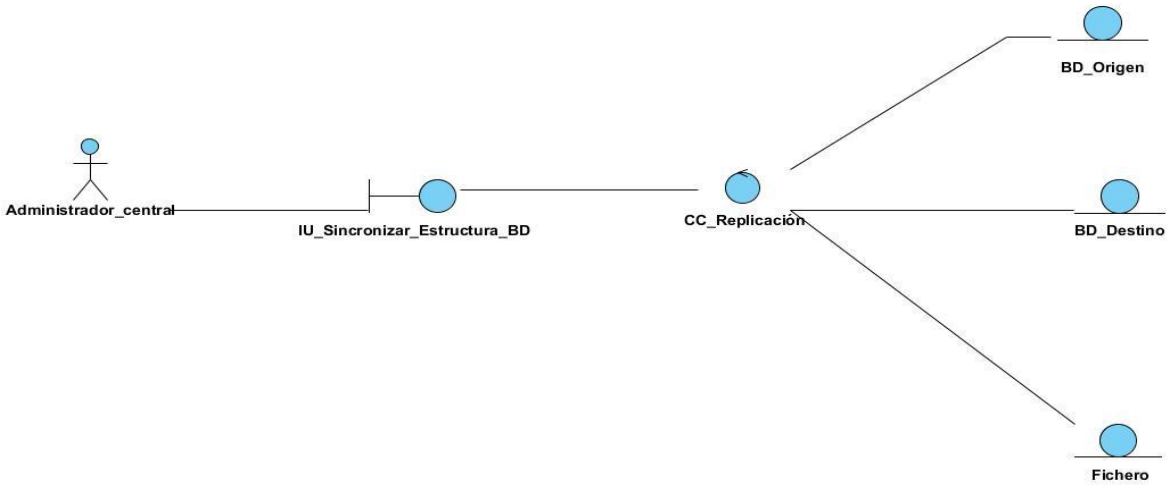


Figura 3.1 DCA Sincronizar estructura de la BD

3.1.2 Diagramas de Colaboración

En el Diagrama de Colaboración (DC) son mostradas las interacciones entre los objetos creando enlaces entre ellos y añadiendo mensajes a esos enlaces. (31)

A continuación se muestra un DC asociado al problema en cuestión, los restantes diagramas se encuentran recogidos en los Anexos.

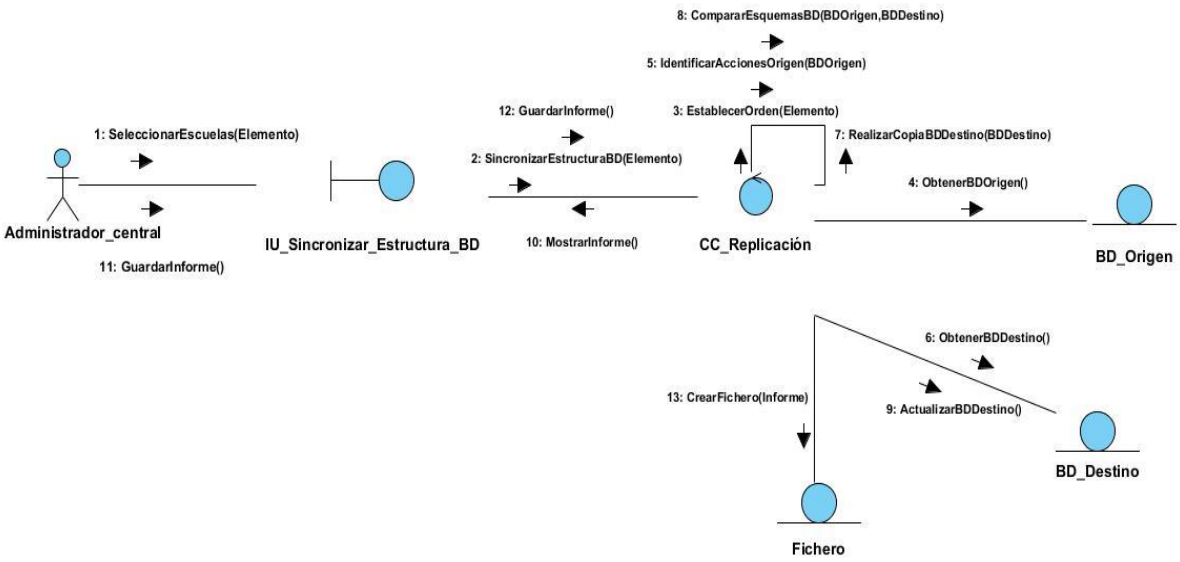


Figura 3.2 DC Sincronizar Estructura de la BD

3.2 Diseño del sistema

3.2.1 Diagramas de Clases del Diseño

El Diagrama de Clases del Diseño (DCD) representa las clases que serán utilizadas dentro del sistema y las relaciones que existen entre ellas.

Las clases representadas en el DCD se clasifican en:

Prototipo	Clases	Descripción
	**Server Page**	Representa la página web que tiene código que se ejecuta en el servidor.
	**Client Page**	Representa a la página web que se ejecuta en el cliente.
	**Form**	Colección de elementos de entrada que son parte de una página cliente.

Tabla 3.2 Clases del diseño

A continuación se muestra un DCD asociado al problema en cuestión, los restantes diagramas se encuentran recogidos en los Anexos.

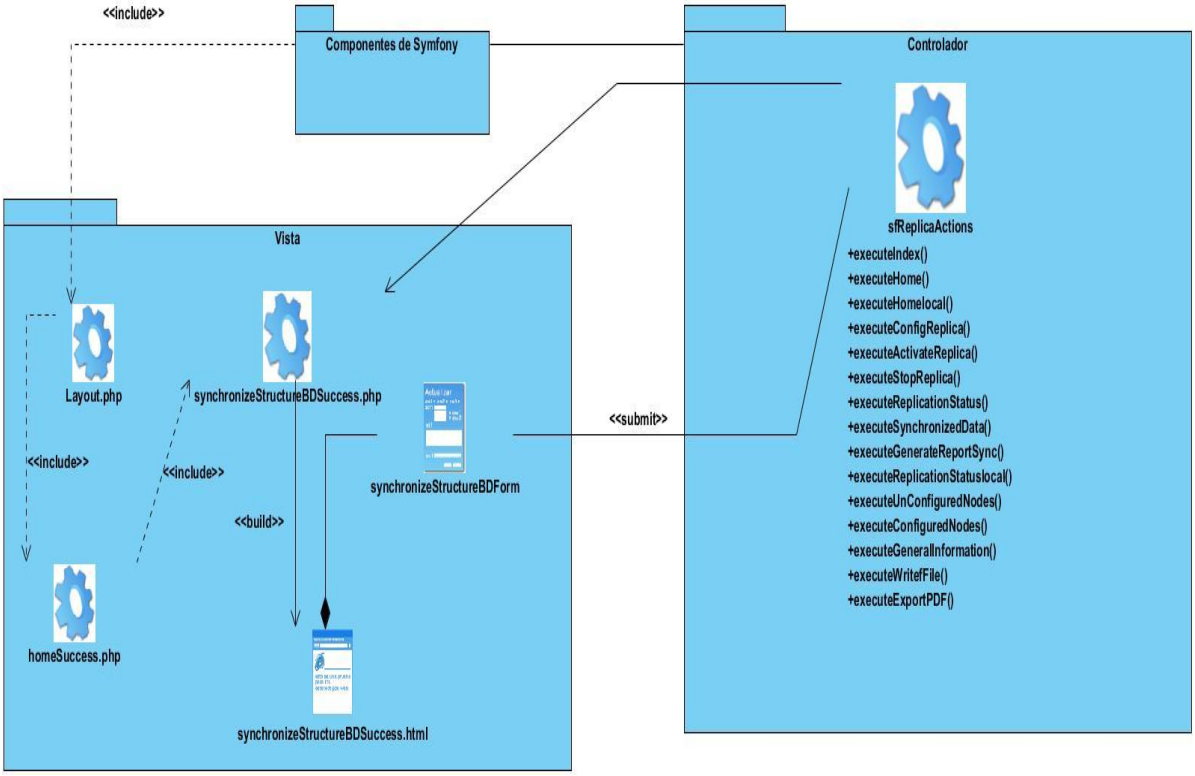


Figura 3.2 DCD Sincronizar Estructura de la BD

3.3 Patrones de Diseño aplicados

Los patrones de diseño permiten describir fragmentos y reutilizar ideas de diseño, basados en la experiencia de otros y que han sido demostrados que funcionan. Estos son soluciones simples a problemas específicos que son comunes del diseño orientado a objetos. (48)

El *framework* Symfony utiliza varios patrones, de los cuales se tuvieron en cuenta algunos de estos durante el diseño del módulo.

Los patrones GRASP (Patrones de Principios Generales para Asignar Responsabilidades) ayudan a comprender el diseño de objetos esencial y aplican el razonamiento para el diseño de una forma sistemática, racional y explicable. Este enfoque para la comprensión y utilización de los principios de diseño se basa en los patrones de asignación de responsabilidades. (48)

Experto: es utilizado para mapear la BD, realizando su capa de abstracción en el modelo, encapsulando toda la lógica de los datos y generando las clases con todas las funcionalidades comunes de las entidades.

Creador: es utilizado en la clase *sfReplicaActions* donde se encuentran definidas las acciones del sistema y se ejecutan cada una de ellas. En esta se crean los objetos de las clases que representan las entidades, siendo esta la creadora de dichas entidades.

Alta Cohesión: ejemplo de su utilización se muestra en la clase *sfReplicaActions*, la cual está compuesta por varias funcionalidades relacionadas, siendo responsable de definir las acciones para las plantillas y colaborar con otras para realizar diferentes operaciones e instanciar objetos.

Bajo Acoplamiento: está presente ya que la clase *sfReplicaActions* hereda únicamente de *sfActions* alcanzando el bajo acoplamiento, las clases que implementan la lógica del negocio y de acceso a datos se encuentran en el modelo, las cuales no tienen asociaciones con las vistas o el controlador, lo que garantiza que la dependencia sea baja.

Controlador: todas las peticiones son manejadas por la clase *sfReplicaActions* que es el punto de entrada único del módulo.

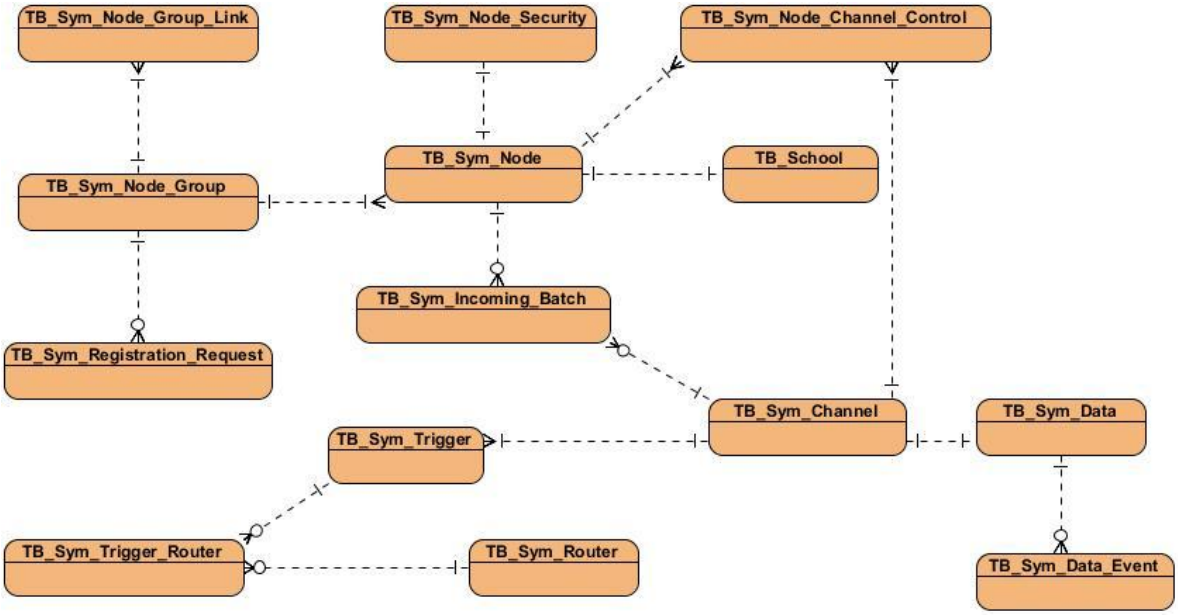
El patrón Modelo-Vista-Controlador (MVC) separa los datos de la aplicación, la interfaz de usuario y la lógica en tres componentes distintos. La vista dónde es guardado el código de la presentación. El modelo que guarda el código de manipulación de los datos. La lógica de procesamiento de las peticiones constituye el controlador. El uso de este patrón ayuda a que el desarrollo del sistema sea rápido y sencillo. (38)

Modelo: se encuentran las clases que son generadas de forma automática según la estructura de la BD.

Vista: es la responsable de producir las páginas que son mostradas como resultado de las acciones, donde se encuentra el *layout*, el cual es común para todas las páginas de la aplicación.

Controlador: se encuentran las acciones que son el núcleo del sistema, ya que contienen la lógica de la aplicación. Estas emplean el modelo y precisan las variables para la vista.

3.4 Diseño de la BD



Las descripciones de las tablas se encuentran en los Anexos.

3.5 Conclusiones Parciales

Fue realizado el análisis y el diseño de la versión 2.0 del módulo Réplica de la Plataforma Educativa Zera. Como parte del análisis fueron generados los DCA y los DC para cada CU del sistema. En el diseño quedaron explícitos los DCD especificando las clases que deben emplearse en la realización del sistema. Fueron aplicados varios patrones de diseño que permitirán el desarrollo rápido y sencillo del módulo.

Capítulo VI: Validación de la propuesta

Introducción

En el presente capítulo es realizada a través de métricas la validación de los requerimientos y el diseño realizado. Tiene como objetivo demostrar que la definición de los requisitos define realmente el sistema que el cliente desea y que en el diseño fueron aplicados correctamente los patrones, garantizando la calidad del diseño orientado a objetos.

4.1 Métricas para evaluar la calidad de la especificación de los requisitos

La validación de los requerimientos demuestra que estos realmente definen el sistema que desea el cliente. Es importante pues los errores existentes en ellos pueden producir importantes coste al ser descubiertos durante el desarrollo o después que el sistema esté en uso. Un cambio en los requisitos significa que el diseño y la implementación del sistema también deben cambiar. (47)

Las métricas de software tienen un papel decisivo en la obtención de un producto de alta calidad porque determinan mediante estadísticas basadas en la experiencia, el avance del software y el cumplimiento de parámetros requeridos. (49)

Según el estándar IEEE 830, 1998 se considera que una especificación es de calidad si cumple con las siguientes características: (50)

- Especificación correcta: un conjunto de requisitos es correcto únicamente si todos los requisitos contenidos representan algo que es requerido para la construcción del sistema y no hay errores que afecten el diseño.
- Especificación no ambigua: un requisito es no ambiguo solo si puede estar sujeto a una única interpretación.
- Especificación completa: solo si, describe todos los requisitos relevantes para el usuario, incluyendo requisitos asociados con funcionalidad, desempeño, restricciones de diseño, atributos o interfaces externas.
- Especificación consistente: siempre y cuando ningún subconjunto de requisitos descrito dentro de ella que esté en conflicto con cualquier otro.
- Especificación verificable: se considera que una especificación es verificable si lo son cada uno de los requisitos constituyentes. Se considera que un requisito individual es verificable si existe un proceso acotado (en plazo y

presupuesto) que permita determinar que el sistema construido satisface lo descrito en el propio requisito.

- Especificación modificable: si su estructura es tal que permite realizar cambios sobre los requisitos que contiene de forma sencilla, completa y consistente, manteniendo la estructura inicial del conjunto.
- Especificación trazable: una especificación es trazable si el origen de cada requisito individual está claro y existe algún mecanismo que permita seguir el impacto de dicho requisito a lo largo del resto de las actividades del ciclo productivo.

4.1.1 Validación de los requisitos del software

El proceso de validación fue realizado a partir de los resultados obtenidos de las revisiones realizadas por miembros del equipo de calidad de la Plataforma Educativa Zera. Del resultado de las revisiones fueron obtenidos los valores de las variables, para a través de cálculos determinar el grado de aplicación de las métricas en los requisitos.

Las variables que se emplean para los cálculos se listan a continuación:

R_f: cantidad de requisitos funcionales.

R_{nf}: cantidad de requisitos no funcionales.

R_t: total de requisitos.

Q_1: grado de especificidad de los requisitos.

Q_2: grado de validación de los requisitos.

Q_3: grado de completitud de los requisitos.

Q_4: grado de comprensión de los requisitos.

Q_5: grado de consistencia interna de los requisitos.

$Q_{6:}$ grado de consistencia externa de los requisitos.

$R_{ii:}$ cantidad de requisitos para los que todos los revisores tuvieron interpretaciones idénticas.

R_C: cantidad de requisitos que se han validado como correctos.

R_{nv}: cantidad de requisitos que no se han validado como correctos.

n_A: cantidad de requisitos completos.

n_B: cantidad de requisitos pobremente especificados**.**

R_{bc}: cantidad de requisitos que todos los revisores entienden.

n_u: cantidad de requisitos especificados.

n_n: cantidad de requisitos en conflicto con otros requisitos en la especificación.

n_{ec}: cantidad de requisitos que son consistentes con otros documentos.

E: valor de la estabilidad de los requisitos.

R_m: cantidad de requisitos modificados que se obtienen como la sumatoria de los requisitos insertados, modificados y eliminados.

A continuación se resumen las métricas aplicadas a los requerimientos del módulo Réplica con sus respectivos resultados: (51)

$$R_t = R_f + R_{nf} = 27 + 11 = 38$$

Especificidad

Para determinar la especificidad (ausencia de ambigüedad) de los requisitos se empleó la métrica basada en la consistencia de la interpretación de los revisores para cada requisito. Cuanto más cerca de 1 esté el valor del grado de especificidad de los requisitos, mayor será la consistencia de la interpretación de los revisores para cada requisito y menor será la ambigüedad de la especificación de los requisitos.

$$Q_1 = R_{ii} / R_t = 36 / 38 = 0.95$$

Corrección

El resultado de esta métrica está siempre entre 0 y 1. El valor óptimo de esta métrica es el más cercano a 1 e indica un alto nivel de corrección en la definición de los requisitos.

$$Q_2 = R_c / R_c + R_{nv} = 38 / 38 + 0 = 1$$

Compleción

El resultado de esta métrica está siempre entre 0 y 1. El valor óptimo de esta métrica es el más cercano a 1 e indica un alto nivel de completitud en la definición de los requisitos.

$$Q_3 = n_A / n_A + n_B = 38 / 38 + 0 = 1$$

Comprensión

La comprensión de los requisitos se determinó a partir de la relación que se muestra a continuación. El resultado de esta métrica está siempre entre 0 y 1. El valor óptimo de esta métrica es el más cercano a 1.

$$Q_4 = R_{bc} / R_t = 38 / 38 = 1$$

Consistencia interna

El resultado de esta métrica está siempre entre 0 y 1. El valor óptimo de esta métrica es el más cercano a 1 y expresa que no existen subconjuntos de requisitos contradictorios. El valor óptimo de esta métrica es el más cercano a 1.

$$Q_5 = n_u - n_n / n_u = 38 - 0 / 38 = 1$$

Consistencia externa

El resultado de esta métrica está siempre entre 0 y 1. El valor óptimo de esta métrica es el más cercano a 1 y expresa que ninguno de los requisitos está en contradicción con lo expresado en el documento de especificación de requerimientos de la Plataforma Educativa Zera. El valor óptimo de esta métrica es el más cercano a 1.

$$Q_6 = n_{ec} / R_t = 38 / 38 = 1$$

Estabilidad

Para medir la estabilidad de los requisitos de software se aplicó la métrica propia para realizar esta acción, la cual ofrece valores entre 0 y 1. El mejor valor es el más cercano a 1. La estabilidad de los requisitos se calcula de la siguiente forma:

$$E = R_t - R_m / R_t = 38 - 2 / 38 = 0.95$$

Partiendo de que la mayoría de las interpretaciones de los revisores coincidían una vez aplicadas las métricas quedó demostrado, que las especificaciones de los requisitos presentan un alto grado de claridad, ya que los requisitos responden a una única interpretación, pues los valores obtenidos son cercano a 1 y mientras más cercano estén estos valores a 1 más consistentes están los requisitos.

4.2 Métricas para evaluar el diseño

La validación del diseño se basa en la calidad del diseño orientado a objetos y la aplicación de los patrones de diseño. (49) Teniendo en cuenta que este estudio brinda un esquema sencillo de implementar, de las métricas existentes para la validación del diseño fueron seleccionadas la Métrica orientada a clases (Tamaño de clases) y Árbol de Profundidad de Herencia (APH).

4.2.1 Tamaño de clases (TC)

El TC se puede determinar empleando métricas para saber el número total de operaciones (tanto operaciones heredadas como privadas de la instancia) que están encapsuladas dentro de la clase y encontrando el número de atributos (tanto atributos heredados como atributos privados de la Instancia) que están encapsulados en la clase. Si existen valores grandes de TC éstos estarán demostrando que una clase puede tener demasiada responsabilidad, lo cual reduciría la reutilización de la clase y hará complicada la implementación y las prueba. De forma contraria sucede si los valores de TC son de menor valor. Por otra parte es necesaria una evaluación concreta de las métricas mediante los umbrales[3]. Finalmente se calculan los promedios correspondientes a los diferentes valores para tener una estimación general del sistema. Algunos especialistas plantean umbrales para estas métricas según como se muestra a continuación, los cuales fueron aplicados al diseño. (52)

Umbral	Tamaño
<=20	Pequeño
>20<=30	Medio
>30	Grande

Tabla 4.1 Umbrales

Al aplicar la métrica de TC en la clase presente, en la capa del controlador quedaron recogidos los siguientes datos:

Clase	Cantidad de atributos	Cantidad de operaciones	Umbral
sfReplicaActions	0	16	Pequeño

Tabla 4.1 Resultados obtenidos

La clase que conforma esta capa se encuentra dentro de la categoría de pequeña lo que demuestra que el diseño del sistema no es complejo.

4.2.2 Árbol de Profundidad de Herencia (APH)

[3] Umbral: valor mínimo de una magnitud a partir de la cual tiene lugar un determinado efecto.

La métrica APH se define como la longitud máxima desde el nodo hasta la raíz del árbol. A medida que crece el APH, es más probable que las clases de niveles inferiores hereden muchos métodos. Esto da lugar a posibles dificultades cuando se intenta predecir el comportamiento de una clase. Una jerarquía de clases profunda (con un valor grande de APH) lleva también a una mayor complejidad de diseño. Por el lado positivo, los valores grandes de APH implican que se pueden reutilizar muchos métodos. (52)

La clase presente en la capa controladora *sfReplicaActions* hereda únicamente de la clase *sfActions* de Symfony. De la cual no hereda ninguna otra clase, por tanto el nivel de profundidad de herencia entre las clases del módulo es 1.

A partir de los datos obtenidos al aplicar la métrica de APH se obtuvo que el nivel de profundidad de herencia más alto entre las clases del diseño es 1, por lo que queda demostrado que el diseño no es complejo, ya que existe un bajo acoplamiento entre las clases y no es difícil su mantenimiento, pues el nivel de profundidad de la herencia es el mínimo que puede existir en la relación entre clases, este resultado conlleva a que el diseño realizado tiene calidad.

4.3 Conclusiones Parciales

La aplicación de las métricas para la calidad de la especificación de requisitos, demostró correspondencia con las necesidades de los clientes y los desarrolladores.

Las métricas aplicadas a los elementos del diseño de la versión 2.0 del módulo Réplica, demostraron la calidad del diseño elaborado para su implementación.

Conclusiones Generales

Luego de realizar las investigaciones y estudios requeridos para desarrollar el análisis y diseño de la versión 2.0 del módulo Réplica de la Plataforma Educativa Zera, y una vez concluida la generación de los artefactos necesarios para proporcionar una amplia y detallada documentación sobre lo que debe hacer el sistema, se arriba a las siguientes conclusiones:

- La correcta selección de los métodos de investigación científica definidos en la investigación que se presenta, permitió conocer la teoría que sustenta el objeto de estudio.
- El análisis realizado sobre las tendencias de los procesos de réplica de datos, como parte del estudio del estado del arte, posibilitó conocer las líneas de desarrollo de estos procesos, para perfeccionar el funcionamiento del módulo Réplica de la Plataforma Educativa Zera.
- Las nuevas funcionalidades propuestas para la versión 2.0 del módulo Réplica, brindan una solución a las deficiencias identificadas en la actual versión del módulo.
- La selección de la metodología de desarrollo RUP, facilitó definir los artefactos necesarios para el cumplimiento del objetivo general del estudio que se presenta.
- Los métodos de investigación científica empleados en la investigación, permitieron realizar la correcta selección de la herramienta y lenguaje para el modelado del análisis y diseño de la versión 2.0 del módulo Réplica.
- La validación realizada demostró la calidad y viabilidad de los principales artefactos generados, para la adecuada implementación del módulo Réplica de la Plataforma Educativa Zera.

Recomendaciones

Al equipo de desarrolladores de la Plataforma Educativa Zera, se le realizan las siguientes recomendaciones:

- Continuar perfeccionando el modelado del sistema mediante la actualización de los cambios que sean necesarios durante las etapas de implementación y pruebas.
- Realizar la implementación de la versión 2.0 del módulo Réplica de la Plataforma Educativa Zera, a partir de la propuesta de diseño dada.
- Tener en cuenta el trabajo como material de estudio para aquellas personas que vayan a realizar una aplicación similar.

Bibliografía

1. **Guillermo Vanegas Arrambide.** *Educación Virtual: Nuevas Formas de Socialidad.* México : Universidad Autónoma de Nuevo León.

2. *Sistemas Distribuidos de Computación. Trabajo Investigativo ¨Base de Datos Distribuidas: Replicación¨.* **Valparaiso, Raul Monge.** 2005.

3. **s Marc Rosenberg.** *E-learning strategies for delivering knowledge in the digital age.*

4. Mas adelante . [En línea] [Citado el: 16 de noviembre de 2011.] http://www.masadelante.com/faqs/base-de-datos.

5. Items Cam. [En línea] [Citado el: 16 de noviembre de 2011.] http://www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r53900.PPT.

6. Pergaminovirtual. *Buscador Hospano.* [En línea] [Citado el: 16 de noviembre de 2011.] http://www.pergaminovirtual.com.ar/definicion/Base_de_datos.html.

7. TECNOMAESTROS. *Base de Datos Distribuidas.* [En línea] [Citado el: 18 de noviembre de 2011.]

8. OOCITIES. *BD Distribuidas.* [En línea] [Citado el: 18 de noviembre de 2011.] http://www.oocities.org/mx/alvarofrio/BDDistribuidas.ppt.

9. **Vicente Toledo – Israel Miralles.** *Bases de Datos Distribuidas.*

10. **C.J.DATE.** *INTRODUCCIÓN A LOS SISTEMAS DE BASE DE DATOS.* s.l. : Pearson Educación. SÉPTIMA EDICIÓN.

11. PlanBSur Solutions. *Alta Disponibilidad.* [En línea] [Citado el: 23 de noviembre de 2011.] http://www.planbsur.cl/soluc_altadisponibilidad.htm.

12. *Aplicación para resolución de conflictos en bases de datos que no ofrezcan.* Bogotá DC : Fundación Universitaria San Martin.

13. **Helena Rivas, Iang Yim, Martín Varela, Javier Frank.** *Técnicas de replicación de datos.* 2003.

14. **Ing. Diosnel Quiñones Rosales, Ing. Alain Arias Yanez.** *Diseño del Motor de Procesamiento de Sentencias para el Sistema Chronos.* Granma : Universidad de las Ciencias Informáticas-Facultad Regional de Granma.

15. **Ing Norge Fajardo Vegas, Ing Alexis Palmas Espinosa.** *Sistema de réplica para bases de datos distribuidas en PostgreSQL.*

16. Corporacion Sybven. [En línea] http://www.corporacionsybven.com/portal/index.php?option=com_content&view=article&id=384:replicacion-de-datos&catid=124:conceptos-teoricos.

17. **Microsoft.** msdn. [En línea] [Citado el: 26 de 05 de 2012.] http://msdn.microsoft.com/es-es/library/ms165742%28v=sql.90%29.aspx.

18. MyGnet . [En línea] [Citado el: 02 de 06 de 2012.] http://mygnet.net/articulos/sql/triggers.774.

19. **Gustavo González,Paola Di Yorio,Luis Ignacio Alonzo,Cesar Barreto.** *Tecnologías para Sistemas de Información.* 2003.

20. **Yissel Fernandez Aguiar, Jose Luis Sierra, Jorge Luis Olmedo Flores.** *Réplica de BD del sistema SIGA.*

21. **Rosell Pupo Polanco,Yenier González Pérez,Ing. Joe Del Toro Domínguez.** *Implementación del componente réplica de base de datos para Akademos v2.0.*

22. PGFoundry. [En línea] http://pgfoundry.org/projects/pgcluster.

23. **Mariano Reingart, Emanuel C. Franco.** *Simple Python-Based PostgreSQL replication system.*

24. Programming 4 Us. [En línea] http://mscerts.programming4.us/es/701376.aspx.

25. JumpMind. [En línea] http://www.jumpmind.com.

26. JumpMind 64. [En línea] http://jumpmind.com/blog/64-symmetricds-240-released.

27. **Eric Long, Chris Henson, Mark Hanes, Greg Wilmer.** *SymmetricDS 2 User Guide v 2.4.* 2011.

28. Xtreme Programming. [En línea] [Citado el: 28 de marzo de 2012.] http://www.extremeprogramming.org/.

29. **innova empresarial.** Metodologías de Desarrollo. [En línea] http://www.google.com.cu/url?sa=t&rct=j&q=METODOLOGIA+MSF&source=web&cd=4&ved=0CD8QFjAD&url=http%3A%2F%2Fwww.innovaempresarial.com%2Fdocs%2FMetodologia_Desarrollo.pdf&ei=A3WhT4j9KpSK6QHr-MX6CA&usg=AFQjCNH04zMR6yxNzO65dW-9WYGtineDKg&cad=rja.

30. **G. Figueroa, Roberth, J. Solís, Camilo y A. Cabrera, Armando.** *METODOLOGÍAS TRADICIONALES VS. METODOLOGÍAS ÁGILES.* s.l. : Universidad Técnica Particular de Loja; Escuela de Ciencias en Computación.

31. **Ivar Jacobson,Grady Booch y James Rumbaugh.** *El Proceso Unificado de Desarrollo de Software.*

32. **JUAN PABLO GOMEZ GALLEGO,ING JORGE GALVES.** *FUNDAMENTOS DE LA METODOLOGIA RUPRATIONAL UNIFIED PROCESS.* UNIVERSIDAD TECNOLÓGICA DE PEREIRA : s.n.

33. **BIZAGI PROCESS MODELER.** BIZAGI BPMN 2.0. [En línea] [Citado el: 21 de 4 de 2012.] www.bizagi.com.

34. **Romero, Oscar Casasola.** Programación en Castellano. [En línea] http://www.programacion.com/articulo/introduccion_a_uml_181.

35. **CACERES, ANA MERCEDES.** *Herramienta case " ArgoUML". .* s.l. : UNIVERSIDAD DON BOSCO, 2006.

36. **SIERRA, MARIA.** *INGENIERÍA DEL SOFTWARE I -Trabajando con Visual Paradigm for UML.* s.l. : Univ. Cantabria – Fac. de Ciencias.

37. Linuxcentro. [En línea] http://www.linuxcentro.net/linux/staticpages/index.php?page=CaracteristicasPHP.

38. **Fabien Pontencier, Francois Zaninotto.** *Symfony la guía definitiva.*

39. PostGreSQL. *PostGreSQL.* [En línea] http://www.postgresql.org/about/press/presskit90.html.es.

40. **Perez, Javier Eguíluz.** *Introducción a XHTML.*

41. **Bos, Bert.** *Web Style Sheets.*

42. **Damián Pérez Valdés.** Maestros de la Web. *¿Que es Javascript?* [En línea] http://www.maestrosdelweb.com/editorial/%C2%BFque-es-javascript/.

43. **Jesse James Garret.** *AJAX un nuevo acercamiento a Aplicaciones Web.*

44. Tecnoretales. Introducción al ORM. [En línea] http://www.tecnoretales.com/programacion/que-es-doctrine-orm.

45. DOCTRINE-PROJECT.ORG. *Doctrine - Object Relational Mapper Documentation (2.0).* [En línea] http://www.doctrine-project.org/projects/orm/2.0/docs/en.

46. **Josep Silva, Jose A Carsi, Isidro Ramos.** *Análisis Teórico-Exeperimental de Criterios de Compracion de Esquemas Conceptuales Orientados a Objetos.* Valencia : Universidad Politécnica de Valencia, 2005.

47. **SOMMERVILLE, IAN.** *Ingeniería del software.* Séptima edición..

48. **Larman, Craig.** *UML y Patrones.*

49. **Pressman, Roger S.** *Ingeniería del Software. Un enfoque prático.*

50. **IEEE Std 830-1998.** *Recommended Practice for Software Requirements Specifications -Description.*

51. **Alan Davis, Scott Overmyer, Kathleen Jordan, Joseph Caruso, Fatma Dandashi,Anhtuan Dinh, Gary Kincaid, Glen Ledeboer, Patricia Reynolds, Pradip Srimani,Anh Ta, and Mary Theofanos.** *Identifying and measuring quality in software requirements specification.* California : s.n.

52. **Pablo Ariel Negro, Dra Roxana Giandini.** *Umbrales para Métricas Orientadas a Objetos.*

53. **Eric Long, Chris Henson.** *SymmetricDS User Guide v 1.0.* 2008.

54. [En línea] http://symmetricds.codehaus.org.

55. **Sergio Alan Flores,Dr Nicardo Farias Mendoza, Msc Armando Roman.** *Modelo de calidad para la microempresa basado en Microsoft.* Colima : s.n.

56. **Bert Bos.** Tutorial introductorio de CSS. [En línea] http://www.w3.org/Style/Examples/011/firstcss.

57. Symfony y el libro Doctrine. Trabajando con los datos. [En línea] http://www.symfony-project.org/doctrine/1_2/es/06-Working-With-Data.

58. **C. Olarte.** Bases de Datos Distribuidas. [En línea]

59. *Replication Strategies: Data Migration, Distribution and Synchronization.*

60. **Grupo de Bases de Datos Avanzadas.** *Fundamentos de Bases de Datos Distribuidas.* Madrid : Univ. Carlos III de Madrid.

61. **Dolores R. Wallace and Laura M. Ippolito.** *Verifying and Validating Software Requirements Specifications.*

62. **Macedo, Roberto Dircio Palacios.** JDBC: Java Database Connectivity. [En línea] [Citado el: 26 de 05 de 2012.] http://ict.udlap/mx/people.

Glosario de Términos

Aplicación: se refiere al programa que se está ejecutando y a los archivos y bases de datos con los que se trabaja.

Configuración: conjunto de variables que controlan la operación general de una aplicación, dichas opciones generalmente son cargadas en su inicio y en algunos casos se deberá reiniciar para poder ver los cambios.

Esquema de BD: describe la estructura de una base de datos.

Identificador: los identificadores son elementos textuales que nombran entidades.

Llave foránea: identifica una columna o grupo de columnas en una tabla, que se refiere a una columna o grupo de columnas en otra tabla.

Plataforma Educativa: es una herramienta virtual que brinda la capacidad de interactuar con uno o varios usuarios con fines pedagógicos.

Ruta: se refiere a la dirección en que serán enviados los datos.

Trigger: evento que se ejecuta cuando se cumple una condición establecida al realizar una operación de inserción (INSERT), actualización (UPDATE) o borrado (DELETE) en la base de datos.

Anexos

Descripciones de los CU

Tabla A.1 CU Activar Réplica

Objetivo	Activarlos procesos de réplica y sincronización de datos entre los servidores.	
Actores	Administrador local (Administrador central): Activar la réplica de una escuela.	
Resumen	El actor selecciona la opción para activar la réplica que permite inicializar los procesos de réplica y sincronización entre los servidores. El sistema inicia la réplica y sincronización entre los servidores. Marca la escuela como activa. El caso de uso termina.	
Complejidad	Alta	
Prioridad	Critico	
Precondiciones	Debe haberse generado el escritorio de trabajo del usuario autenticado. El servidor central debe estar configurado. El servidor local debe estar configurado y desactivado la replicación. Para activar la réplica, el actor debe ser el responsable temporal de la acción.	
Poscondiciones	Se inicializan los procesos de réplica y sincronización para una escuela.	
Flujo de eventos		
Flujo básico Sección 1 ¨Activar la Réplica¨		
	Actor	**Sistema**

1.	El actor selecciona la opción que le permite activar la réplica.	
2.		Si el actor es Administrador local: ***Ver Sección 2: ¨Activar réplica local¨***
3.		Si el actor es Administrador central, identifica la escuela seleccionada.
4.		Comprueba que exista el fichero de configuración inicial para la escuela en la dirección correspondiente del SymmetricDS 2.4.
5.		*Crea un fichero con el comando necesario para inicializar los procesos de réplica y sincronización de datos.*
6.		Muestra un mensaje informativo.
7.		El caso de uso termina.
Sección 2 ¨Activar réplica local¨		
1	En la vista principal del módulo Réplica el actor selecciona la opción para activar la réplica de la escuela.	
2		Identifica la escuela a la que pertenece el actor.
3		Regresa al paso 4 de la *Sección 1 ¨Activar Réplica¨*
Relaciones	**CU Incluidos**	No incluye ningún CU.

	CU Extendidos	No extiende ningún CU.

Tabla A.2 Desactivar Réplica

Objetivo	Detener los procesos de réplica y sincronización de datos entre los servidores.	
Actores	Administrador local (Administrador central): Desactivar la réplica de una escuela.	
Resumen	El actor selecciona la opción para desactivar la réplica que permite detener los procesos de réplica y sincronización entre los servidores. El sistema detiene la réplica y sincronización entre los servidores. Marca la escuela como inactiva. El caso de uso termina.	
Complejidad	Alta	
Prioridad	Opcional	
Precondiciones	Debe haberse generado el escritorio de trabajo del usuario autenticado. El servidor central debe estar configurado. El servidor local debe estar configurado y activado la replicación. Para detener la réplica, el actor debe ser el responsable temporal de la acción.	
Poscondiciones	Se detienen los procesos de réplica y sincronización para una escuela.	
Flujo de eventos		
Flujo básico Sección 1 ¨Desactivar Réplica¨		
	Actor	**Sistema**
1.	El actor selecciona la opción que le permite desactivar la réplica.	

2.		Si el actor es Administrador local: ***Ver Sección 2: ¨Desactivar Réplica local¨***
3.		Si el actor es Administrador central, identifica la escuela seleccionada.
4.		Crea un fichero con los comandos necesarios para detener la réplica y sincronización de la escuela.
5.		Marca la escuela como réplica desactivada.
6.		Muestra un mensaje informativo.
7.		El caso de uso termina.
Sección 2 ¨Desactivar Réplica local¨		
1	En la vista principal del módulo Réplica el actor selecciona la opción para desactivar la réplica de la escuela.	
2		Identifica la escuela a la que pertenece el actor.
3		Regresa al paso 4 de la *Sección 1: ¨Desactivar Réplica¨*

Relaciones	**CU Incluidos**	No incluye ningún CU.
	CU Extendidos	No extiende ningún CU.

Tabla A.3 Configurar ambiente de replicación

Objetivo	Configurar el ambiente inicial de replicación en los servidores.

Actores	Administrador central: configurar ambiente de replicación.
Resumen	El caso de uso se inicia cuando el Administrador central selecciona la opción Configurar, la cual permite la configuración de los ficheros primarios que permiten iniciar la replicación entre los servidores. Luego el actor selecciona la opción Guardar y el sistema genera el archivo de configuración inicial y el caso de uso termina.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	Debe haberse generado el escritorio de trabajo del usuario autenticado. El servidor central debe estar configurado. Para configurar la réplica de la base de datos, el actor debe ser el responsable temporal de la acción.
Poscondiciones	Se generó un fichero con la configuración inicial para la replicación.
Flujo de eventos	
Flujo básico	

	Actor	**Sistema**
1.	El caso de uso inicia cuando el actor selecciona la opción para configurar una escuela que no esté configurada.	
2.		Muestra los campos para que el usuario introduzca los datos necesarios para la configuración inicial: 1. Usuario (se refiere al usuario para conectarse a la BD)

		2. Contraseña (para el usuario de la BD) 3. Base de Datos (nombre de la base de datos) Y permite: • Cancelar la operación. • Guardar la configuración. • Salir de la vista actual.
3.	Introduce los datos necesarios para la configuración.	
4.	Selecciona la opción para guardar la configuración.	
5.		*Valida los datos.*
6.		*Genera el fichero de configuración inicial para la replicación client.properties. Además, tiene en cuenta al definir los canales que todas las tablas relacionadas por llaves foráneas sean incluidas en el mismo canal y establece niveles de replicación para las tablas según las relaciones existentes entre ellas.*
7.		Registra la escuela en el servidor central y asigna el sistema de autenticación para que este acceda al servidor central.
8.		La escuela es registrada como configurada y es eliminada del listado de las no configuradas.

9.		Muestra un mensaje informativo.
10.		El caso de uso termina.
Flujos alternos		
*.a El actor selecciona la opción para cancelar la operación.		
	Actor	**Sistema**
*.a.1		Elimina los datos entrados por el actor.
*.a.2		Muestra un mensaje de información.
*.a.3		Regresa a la vista anterior.
*.a.4		El caso de uso termina.
*.b El actor selecciona la opción para salir de la vista actual.		
*.b.1		Sale de la vista actual.
*.b.2		Muestra la vista seleccionada.
*.b.3		El caso de uso termina.
5. a Existen campos incompletos.		
	Actor	**Sistema**
5.a.1		Muestra un mensaje informativo.
5.a.2		Muestra un indicador sobre cada uno de los campos vacíos.
5.a.3		Regresa al paso 2 del Flujo básico
5. b Existen datos incorrectos.		
	Actor	**Sistema**

5.b.1		Muestra un mensaje informativo.
5.b.2		Muestra un indicador sobre cada uno de los campos vacíos.
5.b.3		Regresa al paso 2 del Flujo básico.
Relaciones	**CU Incluidos**	No incluye ningún CU.
	CU Extendidos	No extiende ningún CU.

Tabla A.4 Generar reporte de sincronización

Objetivo	Mostrar y guardar un informe de la réplica y sincronizaciones de los datos entre los servidores.
Actores	Usuario (Profesor, Director, Administrador central, Administrador local): Ver y guardar reporte de sincronización.
Resumen	El caso de uso inicia cuando el actor selecciona la opción ¨Generar Reporte de Sincronización¨. El sistema le permite al actor seleccionar las opciones por las cuales desea filtrar el reporte y luego le muestra la información según las opciones seleccionadas y le permite guardar el informe. El caso de uso termina.
Complejidad	Media
Prioridad	Opcional
Precondiciones	Debe haberse generado el escritorio de trabajo del usuario autenticado. El servidor central debe estar configurado. El servidor local debe estar configurado y activada la replicación.
Poscondiciones	Fue generado y guardado un reporte de sincronización.
Flujo de eventos	
Flujo básico	

	Actor	Sistema
1.	El caso de uso inicia cuando el actor en el menú del módulo Réplica selecciona la opción ¨Generar reporte de sincronización¨.	
2.		Si el actor es Administrador central, brinda la posibilidad de filtrar por: Nombre de la escuela. Estado de la sincronización. Posibilita seleccionar la información de las sincronizaciones que se van amostrar: • Nombre de la escuela. • Fecha. • Hora. • Estado. • Descripción del error. • Datos sincronizados. *Ver ¨CU Mostrar datos sincronizados¨.* Y permite: • Generar reporte. • Cancelar la operación en cualquier momento. • Salir de la vista actual.
3.	Selecciona las opciones de filtrado deseadas.	
4.	Selecciona la opción para generar el reporte.	
5.		*Valida los datos.*

6.		*Genera un informe con todos los datos pedidos por el actor.* Y permite: Guardar el informe mostrado.
7.	Selecciona la opción para guardar el informe mostrado.	
8.		*Genera un archivo y le solicita al usuario que seleccione dónde guardarlo.*
9.	Selecciona la dirección donde desea guardar el archivo.	Guarda el archivo generado en la dirección indicada.
10.		Muestra un mensaje informativo.
11.		El caso de uso termina.
Flujos alternos		
2.a El actor no es Administrador central		
	Actor	**Sistema**
2.a.1		Muestra todas las opciones del paso 2a excepción del filtrado por nombre de la escuela.
2.a.2		Regresa al paso 3.
*. b El actor selecciona la opción para cancelar la operación		
	Actor	Sistema
*.b.1		Elimina los datos entrados por el actor.
*.b.2		Muestra un mensaje de información.
*.b.3		Regresa a la vista anterior.

*.b.4		El caso de uso termina.
*. c El actor selecciona la opción para salir de la vista actual.		
	Actor	**Sistema**
*.c.1		Sale de la vista actual.
*.c.2		Muestra la vista seleccionada.
*.c.3		El caso de uso termina.

Relaciones	CU Incluidos	No incluye ningún CU.
	CU Extendidos	¨CU Mostrar datos sincronizados¨.

Tabla A.5 Mostrar estado de replicación

Objetivo	Revisar en qué estado se encuentra la réplica y sincronización de los datos entre los servidores.
Actores	Usuario (Profesor, Director, Administrador central, Administrador local): Ve el estado de la replicación de una escuela.
Resumen	El caso de uso inicia cuando el actor selecciona la opción ¨Mostrar Estado de Replicación¨ del menú Réplica. El sistema permite que el usuario filtre la información por el nombre de la escuela si es Administrador central, luego muestra un informe sobre el estado de las sincronizaciones. El caso de uso termina.
Complejidad	Media
Prioridad	Auxiliar
Precondiciones	Debe haberse generado el escritorio de trabajo del usuario autenticado. El servidor central debe estar configurado. El servidor local debe estar configurado y activado la

	replicación.
Poscondiciones	Fue revisado el estado de replicación de la escuela.
Flujo de eventos	
Flujo básico Sección 1: ¨Ver estado de replicación de las escuelas¨	

	Actor	Sistema
1.	El caso de uso inicia cuando el actor en el menú del módulo Réplica selecciona la opción ¨Mostrar estado de replicación¨.	
2.		Si el actor no es Administrador central: ***Ver Sección 2: ¨Ver estado de replicación local¨***.
3.		Brinda la posibilidad de filtrar por: • Nombre de la escuela. • País. • Ciudad. Y permite: • Mostrar estado de replicación de una o varias escuelas. • Cancelar la operación en cualquier momento. • Salir de la vista actual.
4.	Selecciona el nombre de la escuela para ver su estado de replicación.	
5.	Selecciona la opción para mostrar el estado de replicación de la escuela.	
6.		*Valida los datos.*

7.		*Busca el historial de replicación para la escuela seleccionada.*
8.		Muestra un informe con el nombre de la escuela la fecha y la hora de las sincronizaciones realizadas, su estado (Completada o Fallida) coloreando de rojo las fallidas. Además para el caso de las fallidas muestra una breve descripción del error.
9.	Consulta la información mostrada.	
10.		El caso de uso termina.
Flujos alternos		
*. a El actor selecciona la opción para cancelar la operación.		
	Actor	**Sistema**
*.a.1		Elimina los datos entrados por el actor.
*.a.2		Muestra un mensaje de información.
*.a.3		Regresa a la vista anterior.
*.a.4		El caso de uso termina.
*. b El actor selecciona la opción para salir de la vista actual.		
	Actor	**Sistema**
*.b.1		Sale de la vista actual.
*.b.2		Muestra la vista seleccionada.
*.b.3		El caso de uso termina.
6. a Existen campos incompletos.		

	Actor	Sistema
6.a.1		Muestra un mensaje informativo.
6.a.2		Muestra un indicador sobre cada uno de los campos vacíos.
6.a.3		Regresa al paso 2 de la *Sección 1: ¨Ver estado de replicación de las escuelas¨*
Sección 2: "Ver estado de replicación local"		
Flujo básico		
1.	Selecciona la opción para ver el estado de replicación de su escuela.	
2.		Muestra un informe con la fecha y hora de las sincronizaciones realizadas, su estado (Completada o Fallida) coloreando de rojo las fallidas. Además para el caso de las fallidas muestra una breve descripción del error. Y permite: ➢ Salir de la vista actual.
3.		El caso de uso termina.
*. b El actor selecciona la opción para salir de la vista actual.		
	Actor	**Sistema**
*.b.1		Sale de la vista actual.
*.b.2		Muestra la vista seleccionada.
*.b.3		El caso de uso termina.

Relaciones	**CU Incluidos**	No incluye ningún CU.
	CU Extendidos	No extiende ningún CU.

Tabla A.6 Sincronizar estructura de la BD

Objetivo	Sincronizar los cambios realizados en la estructura de la BD con las BD de los servidores locales.	
Actores	Administrador central.	
Resumen	El actor selecciona la opción de ¨Sincronizar Estructura de la BD¨, el usuario debe seleccionar una o varias BD destinos, luego de que este realiza la selección, el sistema realiza una copia de seguridad de cada uno de los destinos y en orden realiza la sincronización. Al terminar cada una de ellas genera un informe.	
Complejidad	Alta	
Prioridad	Crítico	
Precondiciones	Debe haberse generado el escritorio de trabajo del usuario autenticado. El servidor central debe estar configurado. El actor es el responsable temporal de la acción.	
Poscondiciones	Fueron sincronizadas las BD locales.	
Flujo de eventos		
Flujo básico		
	Actor	**Sistema**
1.	El caso de uso inicia cuando el actor selecciona la opción ¨Sincronizar Estructura de la BD¨.	
2.		Brinda la posibilidad de filtrar por:

		• País. • Ciudad. Muestra un listado con los nombres de las escuelas, para que el actor seleccione las escuelas a las que quiere sincronizar los cambios de la BD. Y permite: 1. Sincronizar las BD. 2. Cancelar la operación. 3. Salir de la vista actual.
3.	Selecciona la escuela o las escuelas a las que desea sincronizar los cambios en la BD.	
4.	Selecciona la opción para sincronizar las BD.	
5.		*Valida los datos.*
6.		*Establece un orden entre las escuelas para ir realizando la sincronización de las BD. Luego identifica todas las acciones realizadas en los elementos del origen.*
7.		*Realiza una copia de seguridad de las BD destinos.*
8.		Inicia el proceso de comparación de esquemas de cada BD sucesivamente. Siguiendo los criterios de comparación, identifica las acciones a realizar sobre los elementos de BD.

9.		Teniendo identificadas las acciones a realizar inicia el proceso de actualización de la BD destino.
10.		Al finalizar con cada una de las BD destinos, genera un informe con el nombre de las escuelas y si fue completada o no la sincronización, especificando en caso de fallida.
11.		Permite que el usuario guarde el informe. 3. Guardar el informe.
12.	Selecciona la opción para guardar el informe.	
13.		Solicita al usuario que seleccione donde guardarlo.
14.	Selecciona la dirección donde guardar el archivo.	
15.		Guarda el archivo generado en la dirección indicada.
16.		Muestra un mensaje informativo.
17.		El caso de uso termina.
Flujos alternos		
***. a El actor selecciona la opción para cancelar la operación.**		

	Actor	**Sistema**
*.a.1		Elimina los datos entrados por el actor.

*.a.2		Muestra un mensaje de información.
*.a.3		Regresa a la vista anterior.
*.a.4		El caso de uso termina.
*. b El actor selecciona la opción para salir de la vista actual.		
	Actor	**Sistema**
*.b.1		Sale de la vista actual.
*.b.2		Muestra la vista seleccionada.
*.b.3		El caso de uso termina.

Relaciones	**CU Incluidos**	No incluye ningún CU.
	CU Extendidos	No extiende ningún CU.