Universidad de las Ciencias Informáticas

Facultad 2

Software para la Interconexión de Redes Aisladas CID-555. Módulo Sincronización de Bases de Datos.

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

Autor:
Kenia Rodríguez Suárez

Tutores:
Ing. Carlos Manuel Hernández Vega
Ing. Yosbel Morales Velázquez

La Habana
Junio, 2013

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas y a la Facultad 2 para que hagan el uso que estimen pertinente con este trabajo.
Para que así conste firmamos la presente a los ____ días del mes de junio del 2013.

_____	_____
Autor:	Tutor:
Kenia Rodríguez Suárez	Carlos Manuel Hernández Vega

DEDICATORIA

Le dedico este trabajo a mi familia que tanto ha hecho por mí. En especial a mi abuelita Mima, a mi hermana Katy y a mis padres Luis y Alejandrina. A mis tíos Nemesio y Ángela, que más que tíos son otros padres para mí. Y a todas mis amistades que estuvieron junto a mí en esta larga carrera.

AGRADECIMIENTOS

A mi familia en primer lugar por darme el apoyo que siempre necesité aunque mis decisiones no fueran siempre las correctas. A mi hermanita grande que siempre me pone por delante de ella y a mi mamita que siempre la secunda y que ha sacrificado todo por mí. A mi papi por todo su apoyo incondicional y por sus buenos consejos de siempre, gracias por el hermoso hábito de leer que me inculcaste. A mi tía Ángela y mi tío Nemesio por ayudarme a soportar la lejanía y por ser mis segundos padres. A mi tía Mayi, que junto a mis primas Leslie y Liset me ayudaron para que yo estuviera hoy donde estoy, mil gracias por todo. A mi abuelita que me dio más de lo que pude desear de una abuela tan maravillosa como es ella. A todos mis primos y primas que siempre me ayudaron en todo y siempre se preocuparon por cómo iban mis estudios. A mis amistades que son muchos, Elizabeth mi amiga desde hace 10 años, gracias por estar ahí siempre que te lo pedí. Otra persona que siempre está ahí para mí, y digo siempre, no importa momento, lugar, ni hora, y al cual ni trabajando 1000 años le pago los mil favores que le debo y los cuales siempre me cobra en 10 CUC, Andrés mi mejor amigo de siempre. Isisdais, por tantas locuras y risas compartidas, Jose Carlos (Tato), mi hijo postizo, por tantas carreras que te hice dar, a Nosdaly (Nana), por tantos consejos. Al mejor profesor que he tenido nunca, más que profesor fue un padre para mí, Abel, gracias por apoyarme siempre y por siempre creer en mí. A mis tutores, Carlos y Yosbel, que me ayudaron en todo y estuvieron siempre a mi disposición. A Vladimir, por ayudarme siempre en todo lo que te pedí. A Josefa (Fefa), por ser otra madre para mí y estar siempre pendiente de cómo me van las cosas, por tenerme siempre en mente, gracias. A Janet, por tanto apoyo y consejos. A Doris, Rosa y Eduardo, por siempre preocuparse, a pesar del poco tiempo en que nos conocemos. A todos los profesores que me ayudaron durante la carrera. Al tribunal por tantos aportes tanto

al desarrollo de la tesis como a mi conocimiento en general. Por último y no por ello menos importante a mi novio, Rafael, que a pesar del poco tiempo que estamos juntos, me ha enseñado que la vida da buenas oportunidades, que sólo hay que aprovecharlas, por tantos buenos momentos y tanto amor, gracias.

Si se me queda alguno por favor perdónenme, gracias por formar parte de mi vida y por ayudarme a convertir mi sueño en un hecho.

RESUMEN

El presente módulo, tiene el objetivo de sincronizar dos bases de datos situadas en redes separadas. Ante la tarea de publicar información en Internet y tratando de evitar que información valiosa se vea amenazada, muchas empresas han decidido adoptar la medida de separar las redes, para comunicar con Internet sólo la red externa con la información a publicar, evitando así la penetración por parte de agentes externos a la red interna con la intención de comprometer la información guardada.

La sincronización de las bases de datos ubicadas en ambas redes es esencial para que la información a publicar sea confiable, de ahí la necesidad de crear un módulo para la replicación y sincronización de las bases de datos ubicadas en ambos servidores.

Palabras claves: replicación, sincronización, bases de datos.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Desde el surgimiento y desarrollo de la informática, siempre ha sido de primordial importancia guardar la información de forma segura y con facilidad de acceso. Hasta nuestros días se ha visto un aumento de las amenazas hacia la disponibilidad y confiabilidad de la información utilizando como vía principal internet. Esto se debe a que con el creciente intercambio de información por parte de instituciones, oficinas, empresas, desde cualquier parte del mundo, ha acentuado el intento de acceso a información por parte de agentes externos. El manejo de información, a veces de forma irresponsable, de empleados de oficina, especialistas y personal en general, hace más fácil el trabajo para estos agentes externos.

A pesar de que se han desarrollado conjuntamente tecnologías tales como los cortafuegos, los sistemas de detección de intrusos, entre otras, con el objetivo de lograr una mayor seguridad, aun así la información y los recursos todavía son vulnerables a las intromisiones externas.

Hasta el momento se ha demostrado que estos métodos no han sido lo suficientemente fuertes como para satisfacer completamente la seguridad en entornos altamente críticos. Con el objetivo de evitar los ataques de intrusos, una de las alternativas más radicales es la separación física de los activos a proteger, que se ha propuesto desde hace años como una solución de alto de nivel de seguridad, con el objetivo de evitar por completo los ataques de intrusos mediante la separación física de las redes.

La separación física de las conexiones de red asegura el control del intercambio de datos, imposibilita la configuración incorrecta del sistema de forma tal que puedan establecerse conexiones accidentales, así como es igualmente imposible para cualquier intruso penetrar una red segura desde el exterior.

En el Ministerio de Informática y Comunicaciones (MIC) la red interna está aislada de la red externa con acceso a Internet, surgió la necesidad de comunicar el servicio de correo institucional ubicado en la red interna con Internet. Para dar respuesta a este problema, la Universidad de las Ciencias Informáticas (UCI) junto al Instituto Central de Investigaciones Digitales (ICID) desarrollaron la solución CID-555: Software para intercomunicación de redes aisladas. El diseño y construcción del dispositivo de hardware se realizó en el ICID y el desarrollo del software en la UCI.

Gracias a la separación física que implementa el CID-555 se impiden automáticamente todas las formas conocidas de ataques en línea: spyware, puertas traseras, ataques a nivel de protocolo TCP (Protocolo de Control de Transmisión)[1], la construcción de túneles, encapsulación de mensajes entre otros.

El principio de la separación física de redes no puede tener debilidades de ninguna manera. El CID-555 además de la seguridad que proporciona se puede personalizar y potenciar sus funcionalidades. Su diseño modular permite la integración de aplicaciones adicionales al sistema, como filtro de datos, protección antivirus, sistemas de autenticación y más.

En un inicio, la solución del CID-555 permite mantener comunicados los servidores de correo. Además, permite usar la pasarela de correos por otras aplicaciones para comunicarse, encapsulando la información en mensajes SMTP (Protocolo simple de transferencia de correo)[2]. Como en el aislamiento físico se delimitan dos zonas de servidores, una interna y otra externa, entre las cuales es necesario comunicar servidores, se puede utilizar la pasarela de correos para comunicarlos.

Las empresas que han adoptado la medida de separación física se han enfrentado a nuevas polémicas al tener un mismo servicio en ambas redes. Cuando esto ocurre se publica en el servidor externo la información del servidor interno, para no comprometer la información que se encuentra en este último. El inconveniente viene siendo mantener sincronizados los datos de ambos servidores para que la información publicada en el servidor externo sea fidedigna y equivalente a la del servidor interno.

Para mantener sincronizada una aplicación desplegada en subredes separadas sería necesario sincronizar sus bases de datos. Aprovechando el funcionamiento de la pasarela de correos CID para enviar la información de una subred a otra, se deriva el siguiente **problema a resolver**: ¿Cómo realizar la sincronización de dos bases de datos ubicadas en redes aisladas físicamente entre sí usando la Pasarela de Correos CID-555?

El objeto de estudio se centrará en: los procesos de réplica y sincronización de las bases de datos.

Se enmarca el campo de acción: en los procesos de réplica y sincronización de dos bases de datos ubicadas en redes aisladas físicamente.

[1] TCP es un protocolo orientado a conexión, es decir, que permite que dos máquinas que están comunicadas controlen el estado de la transmisión de datos.[1]

[2] SMTP es el protocolo estándar que permite la transferencia de correo de un servidor a otro mediante una conexión punto a punto. [1]

Se define como idea a defender: la implementación de un módulo de sincronización de bases de datos integrado al software para la interconexión de redes aisladas que permita mejorar los servicios del centro de datos del MIC.

El objetivo general del trabajo el desarrollo de un módulo a la solución pasarela de correos CID-555 para la sincronización de bases de datos en redes aisladas físicamente.

Desglosando el objetivo general en los siguientes objetivos específicos:

1. La implementación de un servicio que sincronice dos bases de datos, de base de datos principal a base de datos réplica. Implementar usando la técnica de generar un script a partir de los cambios en la base de datos principal.
2. El desarrollo del módulo a la pasarela de correos CID-555 que gestione el servicio de sincronización de base de datos. Controlando el flujo formado por las etapas: generación del archivo de sincronización, empaquetado, intercambio, desempaquetado e importado en la base de datos réplica.

Para el cumplimiento de estos objetivos se propone la realización de las siguientes tareas:

1. Análisis de las herramientas de réplica y sincronización de bases de datos más utilizadas actualmente en la UCI, en Cuba y en el mundo. Seleccionar y especificar los requerimientos a implementar en la presente solución.
2. Análisis de las herramientas y metodologías para el desarrollo de aplicaciones más utilizadas actualmente en la UCI, en Cuba y en el mundo. Seleccionar las que se van a usar durante el desarrollo.
3. Implementación del módulo de sincronización de bases de datos sobre la plataforma existente y elaborar la documentación del desarrollo.
4. Instalación del presente módulo y poner en funcionamiento en el laboratorio de pruebas de la solución integral.
5. Realización de pruebas de sincronización donde se compruebe la transparencia y la seguridad de la solución. Además capturar las métricas de rendimiento. Mostrar los resultados y conclusiones.

Obteniéndose como posibles resultados:

1. Un software que permita la sincronización entre dos bases de datos.
2. Una aplicación que permita el intercambio de datos entre dos redes aisladas físicamente entre sí.

Métodos teóricos:

- Análisis-síntesis: Este método fue utilizado en todo el proceso investigativo, ya que a través de la descomposición del problema en varias partes, se precisan características y particularidades. Cada una de estas particularidades posibilitan un mejor entendimiento y comprensión del problema en cuestión, para luego con los resultados obtenidos del análisis, buscar sus relaciones y similitudes. Además, este método permite profundizar y desglosar toda la información encontrada sobre las soluciones de replicación de datos.

Métodos empíricos:

- Observación: Este método permitirá obtener conocimiento acerca del comportamiento de un sistema de supervisión. Permitirá observar cómo funcionan, cómo se desarrollan, así como, las ventajas que brindan y sus especificidades de acuerdo al lugar donde se implementen, además de incrementar la visión sobre las condiciones en las que se encuentran los servicios de replicación de datos en la universidad.

El presente documento se estructura en 4 capítulos:

Capítulo 1 *"Fundamentación Teórica",* aborda el estudio del arte del tema relacionado con el trabajo que se desarrolla, los conceptos y definiciones investigados durante el estudio de las tendencias actuales en el contexto en el cual se desarrolla este trabajo. Incluye además un estudio de las herramientas utilizadas para el desarrollo del proyecto, así como las tecnologías, herramientas y lenguajes existentes útiles para el proceso de implementación.

Capítulo 2 *"Características del Sistema"*, en este capítulo se realiza una descripción de la solución, así como las características del sistema a desarrollar, el objeto de automatización y la información que se maneja. Se realizará una propuesta de sistema y el modelo de negocio, las especificaciones de requisitos de software, las dependencias reales y relaciones con otras aplicaciones, los requerimientos funcionales y no funcionales y la definición de los casos de uso.

Capítulo 3 *"Diseño del Sistema"*, se realizará la descripción y análisis de la solución que se propone para darle resolución a la problemática planteada, además de abordarse elementos referidos al diseño del sistema, más específicamente el modelo de negocio del sistema.

Capítulo 4 *"Implementación y Pruebas"*, dedicado a la implementación y a las pruebas del software, se reflejarán los diferentes diagramas de despliegue y de componentes, etc., así como los diferentes casos de prueba y los resultados de los mismos.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1. Introducción

En este capítulo se abordan los conceptos y definiciones estudiados durante el estudio de las tendencias actuales en el contexto en el cual se desarrolla este trabajo. Se argumenta sobre el concepto de pasarela de correos, se describe el ciclo completo de funcionamiento de la Pasarela de Correos CID-555 para realizar la integración de la sincronización de bases de datos. Del mismo modo incluye un estudio de las herramientas utilizadas para el desarrollo del proyecto, así como las tecnologías y lenguajes existentes para tales propósitos.

1.2. Bases de Datos

Una base de datos es una colección de informaciones o datos relacionados con un tema particular. Está compuesta por campos, ficha, registros y archivos.

Un campo es la pieza más pequeña de información de la que se compone una base de datos. Esta parte indivisible contiene un único dato.

Los campos se disponen en conjunto llamados registros. Un registro es, en medios electrónicos de almacenamiento, el equivalente a la ficha.

Del mismo modo que las fichas, cada registro está compuesto por los mismos campos y en la misma disposición; sólo cambia el contenido, pero permanecen invariables la longitud y la ubicación de cada uno de los datos en todos los registros.

Esta colección de registros idénticos en cuanto a su formato, se agrupa lo que se denomina una tabla, que es el equivalente del contenedor o fichero manual.

La denominación de tabla se debe a su organización en forma de filas y columnas. Cada fila o renglón contiene los datos de un único registro, dispuestos uno al lado del otro, por lo que, siguiendo hacia abajo, en cada columna se encontrará un determinado campo de cada uno de los registros de la tabla. [2]

Los investigadores, científicos y académicos han encontrado muy útil e importante la utilización de esta herramienta para el desarrollo del conocimiento, ya que han sido utilizadas como fuentes de consulta y de producción.

De esta manera la Ciencia de la Información, ha desarrollado una producción científica importante a nivel mundial, la cual ha utilizado las bases de datos, como repositorio de almacenamiento y difusión de información.

1.3. Replicación de Bases de Datos

La replicación es la operación de transportar idénticamente la información de las tablas deseadas de un nodo a otro mediante la red, posibilitando la corrección, disponibilidad, coordinación y efectividad de los datos en un momento y lugar determinado.

La replicación es un conjunto de tecnologías destinadas a la copia y distribución de datos y objetos de base de datos, desde una a otra, para luego sincronizarlas y mantener su coherencia.

Este proceso permite distribuir datos entre diferentes ubicaciones y entre usuarios remotos o móviles mediante redes locales y de área extensa, conexiones de acceso telefónico, conexiones inalámbricas e Internet.[3]

Replicación también se entiende por un mecanismo que permite mantener copias actualizadas automáticamente de los datos de un servidor de bases de datos en otros. Un proceso de compartir información a fin de garantizar la coherencia entre los recursos redundantes.

Los modelos de replicación son la forma en que se han llevado a la práctica el enfoque de replicación de datos por distintas empresas y entidades que los han adoptado. Se describen como un mecanismo que permite mantener copias actualizadas automáticamente (si fuese el caso) de los datos de un servidor de bases de datos en otros, los mismos servirán como base de la investigación a partir de los elementos más importantes identificados en cada uno de ellos.

Existen dos modelos:

- Sincrónico
- Asincrónico

Actualmente existen dos tipos o entornos de réplicas:

- Multi-Maestro
- Maestro-Esclavo

Estos modelos y entornos, son utilizados en pares y asociativos, es decir, el modelo sincrónico o el asincrónico, aplicados a los entornos de réplica: multi-maestro o el maestro-esclavo.

Sincrónico

El modelo de replicación sincrónico, también conocido como la réplica en tiempo real, aplica cualquier cambio o ejecuta cualquier procedimiento reproducido en todos los sitios que participan en el ambiente de réplica como parte de una sola transacción. Si el procedimiento falla en cualquier sitio, entonces la transacción entera se anula. La réplica sincrónica asegura la consistencia de datos en todos los sitios en tiempo real.

Características del modelo sincrónico:

- Actualiza almacenes de datos al mismo tiempo.
- Cada transacción solamente es aceptada si todos los sistemas implicados en la réplica están conectados y listos para recibirla.
- Muy fiable e ideal para recuperarse ante desastres.
- Alto impacto en la red. Poco escalable y caro.[4]

Asincrónico

La replicación asincrónica, a menudo llamada: almacena y reenvía o *Descarga* y *Recarga*, captura cualquier cambio local, los almacena en una cola y a intervalos regulares, propaga y aplica estos cambios en sitios remotos. Con esta forma de réplica, hay un período de tiempo antes de que todos los sitios alcancen la convergencia de datos.
Este modelo consiste también en copiar la salva para un dispositivo de almacenamiento para luego distribuir la salva por los demás servidores. Esta técnica presenta el inconveniente de que en la mayoría de las ocasiones se consultan datos que tienen semanas de desactualización, además de que el proceso se realiza de forma manual.[4]

Características del modelo asincrónico:

- Las escrituras se hacen en un “maestro” y con el tiempo se propagan a varios “esclavos”.
- Económico, escalable y flexible.
- Mayor probabilidad de pérdida de datos, (suplantación de información en el envío de información desde las salida hasta el destino si no se tiene un buen sistema de seguridad).

Se puede combinar perfectamente con los entornos de réplica, de acuerdo a los intereses de quien lo utilice, teniendo en cuenta los problemas que surgen una vez compuestos. Dichos problemas no dependen explícitamente del entorno, sino del modelo como tal, en este caso el asincrónico, llegando a surgir más inconvenientes que en el sincrónico.[4]
El modelo a utilizar para la sincronización de bases de datos del módulo es el asincrónico, debido a que la aplicación copiará el script con los cambios ocurridos en la base de datos principal en un dispositivo de almacenamiento USB y no directamente en la carpeta local, de donde el Reko lo tomara para hacer los cambios en la base de datos réplica.

Entorno de Replicación:

Multi-Maestro

El entorno Multi-Maestro, permite múltiples sitios, actuando como pares iguales. Cada sitio en un ambiente de réplica de multi- maestro es un sitio de maestro, y cada sitio se comunica con otros sitios maestros.
Permite leer y escribir las consultas que se enviarán a múltiples servidores replicados. Esta capacidad también tiene un considerable impacto en el rendimiento debido a la necesidad de sincronizar los cambios entre los maestros.

Maestro-Esclavo

El entorno de solo lectura o maestro-esclavo (*master-slave*), permite al nodo maestro realizar consultas de lectura y escritura, mientras que los nodos esclavos solo de lectura.[5]

1.4. CID-555: Software para la intercomunicación de redes aisladas

En el transcurso del curso 2011-2012, se realizó la construcción de un dispositivo para automatizar el envío de correo, esta tarea le fue encomendada al Instituto Central de Investigaciones Digitales (ICID).
Para este dispositivo se implementó el Software para la Interconexión de Redes Aisladas Físicamente en su versión 1.0.
El ICID (Instituto Central de Investigaciones Digitales) es la empresa más moderna y experimentada en el tema de la electrónica en Cuba. Incluye facilidades para la producción electrónica y electromecánica, soportado por décadas de experiencia con las más modernas tecnologías. Además, fabrica partes mecánicas de alta calidad para equipos médicos y otros. Con más de 30 años de experiencia en la

fabricación, la planta de circuitos impresos procesa una gran cantidad de códigos diferentes y fabrica pequeños y medianos lotes con alta eficiencia. El sistema de aseguramiento de la calidad de las plantas de producción del ICID está basado en las normas ISO9001-2000.

Debido a esto la Dirección Informática del Ministerio de las Informáticas y las Comunicaciones solicitó la creación del dispositivo de interconexión CID-555.

El dispositivo de conmutación está conectado a ambos servidores por dos cables serie con conector DB9 y dos cables USB. Internamente tiene dos memorias Flash para el almacenamiento de los mensajes. Las memorias Flash son conmutadas entre ambos servidores por relés electromagnéticos, de tal manera que en un instante de tiempo cada memoria está conectada a un único servidor. La comunicación con el dispositivo, para su control, se realiza por las terminales serie. Las interfaces permiten notificar el estado de la copia hacia la memoria por la solución de software y ordenarle conmutar los relés. Para conocer el estado de la copia, el dispositivo implementa un registro para el Servidor Externo. Para realizar la conmutación el dispositivo implementa un segundo registro que al ser escrito acciona los relés y el tiempo de conmutación de las memorias es configurable por el administrador.

Los componentes del dispositivo son:

- 2 Micro controladores que establece la lógica de funcionamiento
- 2 Módulos de comunicación
- 2 Módulos de conmutación
- 5 Relés electromagnéticos (1 para 1 conexión, 4 para 5 conexiones)
- 2 Puertos DB9
- 2 Puertos USB
- 2 Memorias Flash

Los terminales DB9 del dispositivo deben ser conectados correctamente al servidor correspondiente para el correcto funcionamiento del sistema. Los puertos USB pueden conectarse indistintamente uno a cada servidor.

La arquitectura del dispositivo garantiza que no exista comunicación de señales eléctricas entre el módulo Externo y el Interno. El estado se da a conocer mecánicamente por la acción del Relé de estado.[6]

1.5. Sistemas similares

Con el desarrollo de diferentes aplicaciones informáticas para la intromisión de personas externas a los sistemas informáticos, los virus que surgen cada día y demás herramientas para la apropiación de información, las amenazas procedentes de internet están muy lejos de estar bajo control. La seguridad se ha enfocado principalmente en proteger los servicios de comunicación empresarial de los intrusos llamados hackers. Siguiendo el principio de que "el método perfecto para asegurar una red es que esté desconectada", se ha tomado como medida principal el aislamiento físico de las redes con la esperanza de evitar por completo los ataques de intrusos. Durante la investigación se identificaron sistemas con funcionalidades similares a las de la solución, a continuación se presentan algunas de ellas.

Lock-Keepers

Lock-Keepers es una aplicación novedosa de separación física, surgida en el año 2003 y comercializada por Siemens Switzerland. En los últimos años esta aplicación ha sido mejorada para ser más madura y confiable. Debido a esta separación física Lock-Keepers evita automáticamente todos los ataques en línea conocidos: spyware, puertas traseras, los ataques a nivel de protocolo (por ejemplo, como los ataques TCP número de secuencia), construcción de túneles, mensaje encapsulado, entre otros.[7]

La funcionalidad del Lock-Keeper es tan simple como una compuerta, y tan eficaz. Su principio de seguridad se basa en tres pasos:

1. Los datos entrantes desde una red externa se transfiere en el Lock-Keeper. En el transcurso de esta acción, no hay conexión con la red interior. Figura 1.

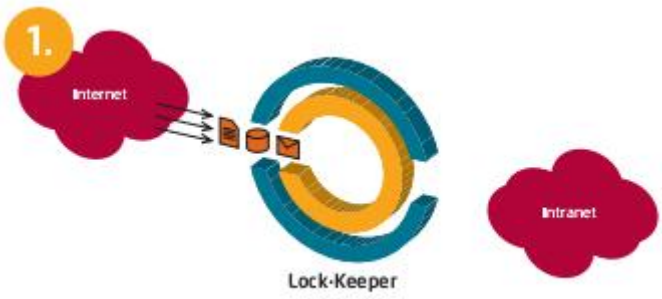


Figura 1. Lock-Keeper. Escenario 1.

2. Una vez que los datos han sido transferidos, el Lock-Keeper se desconecta de la red externa. Los datos están aislados en el sistema del Lock-Keeper. Figura 2.

Figura 2. Lock-Keeper. Escenario 2.

3. Sólo entonces se establece el enlace a la red interna y los datos son transferidos. Figura 3.

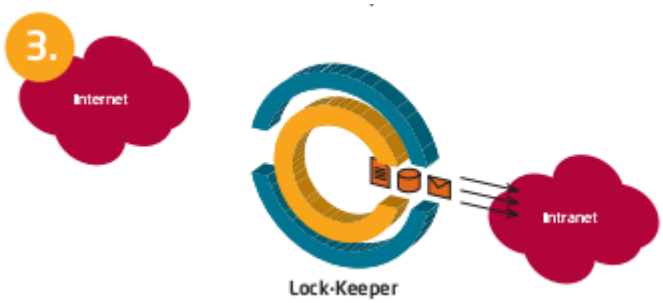


Figura 3. Lock-Keeper. Escenario 3.

El mismo principio se utiliza para transferir datos en la dirección opuesta de la red interna a una externa (por ejemplo, Internet). Gracias a este mecanismo patentado de conmutación, el Lock-Keeper separa completamente dos redes, y a la vez permite datos a transferir entre ellos, mientras puede seguir siendo independiente de cualquier software de aplicaciones que esté involucrado. [7]

CronSQL

CronSQL es una herramienta para la sincronización. Permite definir múltiples sincronizaciones entre diferentes bases de datos de una empresa facilitando la sincronización de ciertos datos entre varios programas e incluso entre páginas web.

CronSQL soporta conexiones a cualquier tipo de base de datos compatible con OLEDB, e incluso es capaz de conectar a través de una interfaz desarrollada en PHP (disponible en el manual de usuario) con bases de datos de servidores web con el acceso externo restringido.
Aunque la interfaz de CronSQL es visual, intuitiva y sencilla, se requieren conocimientos básicos en consultas SQL y sobre el modelo de datos de las bases de datos a sincronizar, para configurar la sincronización. Es por ello que esta es una herramienta muy adecuada para programadores, desarrolladores de páginas web y administradores de bases de datos; también para empresas que dispongan de distintas soluciones de software heterogéneo sin ningún mecanismo de sincronización.
El caso más habitual y para el cual CronSQL fue inicialmente desarrollado, es para sincronizar el software de gestión de una empresa con la base de datos de la página web, de modo que los productos dados de alta en el programa de gestión sean automáticamente llevados a la tienda virtual de la web.[8]

SharePlex for Oracle

SharePlex® for Oracle es una tecnología madura, de alto rendimiento y alta disponibilidad que ofrece una alternativa de bajo costo para otras herramientas de replicación de Oracle. A diferencia de otras soluciones, SharePlex brinda comparación y reparación de datos, integridad de los datos en transacción y funciones de monitorización y alerta, todo en un paquete.
Esta solución de replicación de Oracle asegura la continuidad comercial mientras cumple con los objetivos operacionales de las bases de datos. Le brinda una copia en tiempo real de los datos de producción, sin afectar el rendimiento y la disponibilidad del sistema.
Todas estas tecnologías aunque novedosas tienen una desventaja en común y es que tienen un precio muy elevado y por las condiciones económicas del país no se pueden asumir estos gastos, por lo que eran necesarias otras soluciones más factibles y que contribuyeran a la independencia tecnológica.

1.6. Tecnologías utilizadas

SQLite

SQLite es un sistema de gestión de bases de datos relacional compatible con ACID, contenida en una relativamente pequeña biblioteca escrita en C. El código de SQLite es de dominio público y por tanto, es

libre de ser utilizado para cualquier propósito, comercial o privado. SQLite se encuentra en la actualidad en más aplicaciones de las que se pueden contar, incluyendo varios proyectos de alto perfil.
A diferencia de los sistemas de gestión de bases de datos cliente-servidor, el motor de SQLite no es un proceso independiente con el que el programa principal se comunica. En lugar de eso, la biblioteca SQLite se enlaza con el programa pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de SQLite a través de llamadas simples a subrutinas y funciones. Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. SQLite en general se ejecuta más rápido mientras más memoria que se le asigna. Sin embargo, los resultados por lo general son muy buenos incluso en entornos de poca memoria. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un sólo fichero estándar en la máquina host. Este diseño simple se logra bloqueando todo el fichero de base de datos al principio de cada transacción. SQLite es muy cuidadosamente probados antes de cada lanzamiento y tiene la reputación de ser muy fiable.[9]

Middleware Zeroc-ICE

El motor de comunicaciones de Internet(ICE) es una herramienta orientada a objetos moderna que permite construir aplicaciones distribuidas con el mínimo esfuerzo. ICE permite centrar los esfuerzos en la lógica de aplicación, y se encarga de todas las interacciones con bajo nivel de programación de interfaces de red. Con ICE, no hay necesidad de preocuparse por los detalles como la apertura de las conexiones de red, serialización y deserialización de datos para la transmisión de la red, o volver a intentar los intentos fallidos de conexión.[10]

Las aplicaciones de Ice son adecuadas para su uso en entornos heterogéneos: el cliente y el servidor pueden ser escritos en diferentes lenguajes de programación, puede ejecutarse en diferentes sistemas operativos y arquitecturas de máquina, y puede comunicarse con una variedad de tecnologías de redes.

Mediante Ice, cada objeto tiene una interfaz con un número de operaciones. Las interfaces, operaciones y los tipos de datos que se intercambian entre cliente y servidor se definen mediante el lenguaje Slice.

Slice (*Specification Language for Ice* o Especificación del Lenguaje de Ice) es el mecanismo fundamental de la abstracción para la separación de los objetos de las interfaces de sus

implementaciones. Slice establece un contrato entre el cliente y el servidor que describe los tipos y las interfaces de objetos utilizado por una aplicación. Esta descripción es independiente del lenguaje de implementación, por lo que no importa si el cliente está escrito en el mismo idioma que el servidor. O sea, Slice viene siendo un lenguaje común entre el cliente y el servidor para poder realizar las llamadas a los objetos y métodos entre ambos lenguajes de programación.

ICE está actualmente disponible para C++, Java, .NET, Python, PHP, Ruby, Objective-C, y ActionScript, y es compatible con la mayoría de los sistemas operativos como son Linux, Solaris, Windows, Windows CE, Mac OS X, Unix, GNU/Linux, *BSD, OSX, Symbian OS, J2RE 1.4 o superior y J2ME. [11]

Otros sistemas operativos y lenguajes serán soportados en versiones futuras. Ice también está disponible para dispositivos móviles.[12]

1.6.1. Lenguajes de programación

Es una notación para escribir programas, a través de los cuales es posible la comunicación con el hardware y así dar las órdenes adecuadas para la realización de un determinado proceso. Está definido por una gramática o conjunto de símbolos junto a un grupo de reglas para combinar dichos símbolos que se usan para expresar programas. Constan de un léxico, una sintaxis y una semántica.

El léxico, es un conjunto de símbolos permitidos o vocabulario, la sintaxis, son las reglas que indican cómo realizar las construcciones del lenguaje, la semánticas, son las reglas que permiten determinar el significado de cualquier construcción del lenguaje y los tipos de lenguajes, son los que atendiendo al número de instrucciones necesarias para realizar una tarea específica, se clasifican en dos grandes bloques: bajo nivel y alto nivel.

Los lenguajes de bajo nivel, también llamados lenguajes ensambladores, permiten al programador escribir instrucciones de un programa usando abreviaturas. Es el tipo de lenguaje que cualquier computadora es capaz de entender. Se dice que los programas escritos en forma de ceros y unos están en lenguaje de máquina, porque esa es la versión del programa que la computadora realmente lee y sigue. [10]

Un lenguaje de alto nivel permite al programador escribir las instrucciones de un programa utilizando palabras o expresiones sintácticas. Son lenguajes de programación que se asemejan a las lenguas humanas usando palabras y frases fáciles de entender. Estos lenguajes son independientes de la arquitectura física de la computadora, permiten usar los mismos programas en computadoras de

diferentes arquitecturas (portabilidad), y no es necesario conocer el hardware específico de la máquina. La ejecución de un programa en este lenguaje, requiere de una traducción del mismo al lenguaje de la computadora donde va a ser ejecutado. Utilizan notaciones cercanas a las usadas por las personas en un determinado ámbito. Se suelen incluir instrucciones potentes de uso frecuente que son ofrecidas por el lenguaje de programación.[10]

En un lenguaje de bajo nivel cada instrucción corresponde a una acción ejecutable por el ordenador, mientras que en los lenguajes de alto nivel una instrucción suele corresponder a varias acciones. [10]

Para el desarrollo de la aplicación se utilizarán lenguajes de alto nivel.

Python

Es un lenguaje interpretado o de script, con tipado dinámico, fuertemente tipado multiplataforma y orientado a objetos. Python es un lenguaje de alto nivel que permite escribir código con una alta claridad y legibilidad, permitiendo así un rápido aprendizaje del mismo. La legibilidad permitirá en futuros accesos al código una clara comprensión de lo antes implementado, haciendo más fácil el mantenimiento de las aplicaciones.

Su biblioteca estándar es muy amplia, conteniendo funcionalidades de gran ayuda desde el más bajo nivel hasta el más alto, facilitándole al programador la implementación de aplicaciones sin la necesidad de recurrir continuamente a bibliotecas externas. Además dispone de una extensa colección de bibliotecas libres disponibles en la mayoría de los repositorios de los sistemas GNU\Linux.

Este lenguaje permite ser extendido, haciéndolo mucho más favorable para su uso, pues en caso de existir alguna región crítica del proyecto escrita en Python que no sea la más recomendable u óptima, puede ser implementada en C/C++ y compilada de modo que sea accesible desde el intérprete de Python.

El intérprete de Python funciona en diversos sistemas operativos, tales como: GNU\Linux, Mac OS X y Windows. Debido a la portabilidad de su código; al programar las aplicaciones sin utilizar bibliotecas propias a los sistemas operativos, es posible ejecutarlas en cualquier plataforma sobre la que exista el intérprete de Python.

Teniendo en cuenta que para el desarrollo de los driver que permitirán la sincronización en el hardware al cual se le implementará el software, el lenguaje utilizado fue Python, y además por todas las funcionalidades que brindan para la comodidad del desarrollo, y para seguir la misma línea, se propone como lenguaje de programación.[13]

PHP (Hypertext Preprocessor)[3]

PHP es un lenguaje de scripting que permite la generación dinámica de contenidos en un servidor web. Entre sus principales características cabe destacar su potencia, su alto rendimiento, su facilidad de aprendizaje y su escasez de consumo de recursos.

PHP es de los lenguajes de lado servidor más extendido en la web. Es un lenguaje que ha tenido una gran aceptación en la comunidad de desarrolladores, debido a la potencia y simplicidad que lo caracterizan, así como al soporte generalizado en la mayoría de los servidores de *hosting*.

PHP permite embeber sus pequeños fragmentos de código dentro de la página HTML (*HyperText Markup Language* o Lenguaje de Marcas de Hipertexto)[4] y realizar determinadas acciones de una forma fácil y eficaz, combinándose con HTML. Es decir, con PHP se escriben scripts dentro del código HTML. Por otra parte, PHP ofrece múltiples funciones para la explotación de bases de datos de una manera llana, sin complicaciones.

El código fuente escrito en PHP es invisible al navegador web y al cliente ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable. Posee una amplia documentación en su sitio web oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda. Permite aplicar técnicas de programación orientada a objetos. Además es libre, por lo que se presenta como una alternativa de fácil acceso para todos.[10]

1.6.2. Herramientas de software para el desarrollo de la solución

Herramienta CASE

Dentro de las herramientas claves en el desarrollo de aplicaciones informáticas se encuentran las herramientas de Ingeniería de Software Asistida por Ordenador (CASE), las cuales son las encargadas de ayudar en el ciclo de desarrollo, con el fin de aumentar la productividad y reduciendo el coste en términos de tiempo y dinero. En el ciclo de desarrollo pueden ayudar en el proceso de diseño del proyecto, en el

[3] PHP o Procesador de Hipertexto es un lenguaje script (no se compila para conseguir códigos máquina si no que existe un intérprete que lee el código y se encarga de ejecutar las instrucciones que contiene éste código), para el desarrollo de páginas web dinámicas del lado del servidor.[14]

4 HTML es un lenguaje de programación que se utiliza para el desarrollo de páginas de Internet.[15]

cálculo de costes, pueden implementar una parte del código, compilación automática y documentación. En el presente trabajo se utilizará Visual Paradigm para UML.[10]

Visual Paradigm para UML

Una de las herramientas CASE más usadas es la suite creada por Visual Paradigm International (VPI). VPI es un proveedor de soluciones informáticas que incluye organizaciones para desarrollar aplicaciones de calidad, rápidas y baratas. Sus soluciones se enfocan en eliminar la complejidad, aumentando así la productividad y disminuyendo el tiempo de desarrollo de las aplicaciones informáticas.

La suite VisualParadigm está compuesta por productos que facilitan a las organizaciones la visualización y diseño de diagramas, así como su integración con lenguajes de programación y gestores de bases de datos. Es independiente de las plataformas, soportando varios entornos integrados de desarrollo, entre las que se pueden mencionar: Microsoft Visual Studio, Eclipse y Java.

La herramienta CASE Visual Paradigm fue escogida para desarrollo de la aplicación debido a que utiliza UML como lenguaje de modelado, agiliza la creación de los diferentes diagramas definidos en la metodología RUP, se integra con EasyEclipse, se puede utilizar en sistemas operativos GNU/Linux y genera una excelente documentación en varios formatos como jpg, html y pdf.[10]

Modelo de negocio

Business Process Modeling Notation o BPMN (En español Notación para el Modelado de Procesos de Negocio), es una notación gráfica estandarizada que permite el modelado de procesos de negocio, en un formato de flujo de trabajo (*workflow*). BPMN fue inicialmente desarrollada por la organización *Business Process Management Initiative* (BPMI), y es actualmente mantenida por el OMG (*Object Management Group*), luego de la fusión de las dos organizaciones en el año 2005.

El objetivo primario del lenguaje estándar BPMN fue proveer una notación que sea legible y entendible para todos los usuarios de negocios, desde los analistas que realizan el diseño inicial de los procesos y los responsables de desarrollar la tecnología que ejecutará estos procesos, hasta los gerentes de negocios encargados de administrar y realizar el monitoreo de los procesos. BPMN define un modelo de procesos de negocio basándose en diagramas de flujo. Un modelo de procesos de negocio, es una red de objetos gráficos que representan las actividades (por ejemplo tareas) y los controles de flujo que definen su orden de ejecución. [16]

Actualmente BPMN es el mejor lenguaje para describir paso a paso los procesos de negocios.

REKO

REKO es un software de réplica de datos entre bases de datos desarrollado en la Universidad de las Ciencias Informáticas que pretende cubrir las principales necesidades relacionadas con la distribución de datos entre los gestores más populares entre los que se encuentra PostgreSQL, como la protección, recuperación, sincronización de datos, transferencia de datos entre diversas localizaciones o la centralización de la información en una única localización. Ha sido diseñado para permitir la réplica y sincronización de datos con conexión y sin conexión, la selección de los datos a replicar y la definición de filtros de replicación. Replica datos entre bases de datos con estructuras heterogéneas, y entre gestores heterogéneos. Fue diseñado e implementado completamente usando herramientas libres y librerías de clases con licencia gratuita.

Principales Funcionalidades:

- Tipos de replicación: Soporta la replicación de transacciones a través de Hibernate y la replicación de acciones a través de triggers.
- Gestores soportados: Se integra actualmente con los gestores de base de datos de Oracle Y PostgreSQL.
- Selección de datos: Provee la facilidad de seleccionar el subconjunto de tablas y datos a ser replicados en la base de datos mediante la definición de filtros aplicables a las tablas y la selección de usuarios propios del gestor.
- Transmisión de datos: La trasferencia se datos de replicación puede realizarse a soporte para replicación sobre TCP/IP, FTP[5], HTTP o por ficheros de forma manual. Los archivos de gran tamaño son enviados por FTP permitiendo resumir la transmisión caso de interrupciones en la red.

[5] (FTP o *File Transfer Protocol*), Protocolo de Transferencia de Archivos.

- Configuración entre nodos: El mecanismo de registro entre nodos de replicación se basa únicamente en un identificador (id) del nodo, lo que permite la abstracción de los datos físicos de cada nodo, como son el IP y el protocolo de comunicación.
- Seguridad: Los nodos de replicación manejan credenciales entre ellos para verificar la autenticidad de los datos transferidos. El envío de estos datos se realiza utilizando protocolos seguros de comunicación como son HTTPS y SSL.
- Monitoreo: Permite conocer el estado de los datos trasmitidos. Puede realizarse en tiempo real a través de la Web así como dar seguimiento al funcionamiento interno del mecanismo.
- Independiente de la plataforma: El software de réplica puede ser instalado en cualquier sistema operativo y no necesita de un ambiente grafico en el mismo para su funcionamiento.
- Tolerancia a fallos: Detecta errores de conexión. Mantiene los datos de réplica en un estado estable en caso de desconexión. Al restablecerse la conexión, automáticamente sincroniza los datos entre las BD.
- Soporte para resolución de conflictos automática y manual.
- Réplica de datos de gran tamaño con capacidad de resumen.
- Capacidad para crear y ajustarse a complejos escenarios de replicación.
- Interfaz de administración de réplica.
- Soporte para programación del momento de captura y envío de los datos de réplica.

Principales Componentes:

- Núcleo: Maneja las configuraciones fundamentales del software y agrupa las principales funcionalidades de procesamiento de información.
- Capturador de Cambios: Captura los cambios que se realizan sobre la base de datos y se los entrega al Distribuidor de Cambios.
- Aplicador de Cambios: Ejecuta sobre la base de datos los cambios que sean replicados.
- Distribuidor de Datos: Determina para dónde debe ser enviado cada cambio realizado en la BD, los envía y se responsabiliza de que cada cambio llegue a su destino.

- BD Local de Réplica: Es utilizada para guardar las configuraciones propias de la réplica, las acciones sobre la BD que han dado conflicto al aplicarse y las acciones o transacciones que no han podido llegar a su destino.
- Consola Web de Administración: Representa la interfaz del software. Permite realizar las configuraciones principales del software como el registro de nodos, configuración de las tablas y datos a replicar, y el monitoreo del funcionamiento del software.
- Servidor JMS: Servidor de Mensajería bajo la especificación *Java Message Service*, es utilizado como punto intermedio en la distribución de la información enviada bajo JMS.
- Servidor FTP: El servidor FTP es utilizado de forma opcional en el funcionamiento del software. La función principal de utilizar un servidor de FTP es para enviar grandes archivos de réplica y que se pueda resumir la *trasmisión* de los mismos en caso de problemas en la conexión.
- Base de Datos: Es la base de datos que se está replicando, el software de réplica envía los cambios que se realizan sobre ella y aplica los cambios que provienen de otros nodos de réplica. Figura 4. [17]

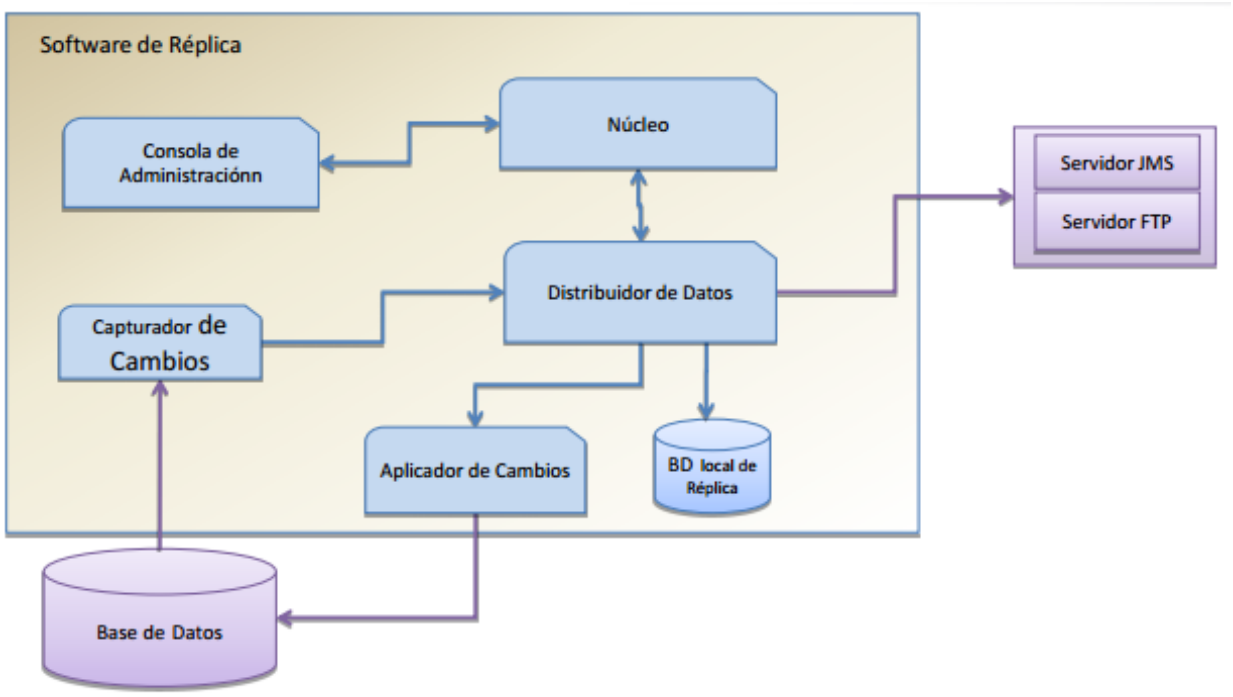


Figura 4. REKO, sus principales componentes.

1.6.3. Metodología de Desarrollo de Software

RUP (*Rational Unified Process*); Proceso Unificado de Rational, es más que un simple proceso; es un marco de trabajo genérico que puede especificarse para una gran variedad de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyecto.

RUP utiliza el Lenguaje Unificado de Modelado para preparar todos los esquemas de un software.

Para llevar a cabo esta investigación, se implementará un gran número de funcionalidades con alta complejidad, por lo que es necesario agrupar las funcionalidades similares de tal manera que el desarrollo de la aplicación sea un proceso comprensible y quede bien documentado. Al ser RUP dirigido por casos de uso, su utilización garantiza que se satisfaga esta necesidad, ya que los casos de uso (CU) guían el proceso de desarrollo, pues los modelos que se obtienen como resultado de los diferentes flujos de trabajo, representan su realización.

Igualmente se necesita tener una visión del sistema completo, realizando primeramente los CU arquitectónicamente significativos, de forma que constituyan los cimientos del sistema necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente. RUP es centrado en la arquitectura, característica que se encuentra en total concordancia con esta necesidad.

Se considera que la utilización de RUP puede ser de suma importancia para el desarrollo de la aplicación de forma eficiente y con alta productividad debido a que define qué se tiene que hacer, cómo y quién lo hace en cada momento del proceso de desarrollo.

También es política del departamento de Seguridad Informática del centro de Telemática la utilización de esta metodología de desarrollo para el proceso de implementación de todos los productos de software.[10]

1.6.4. Entorno de Desarrollo Integrado

Una de las herramientas que desempeña un papel importante en el desarrollo de soluciones informáticas son los Entornos de Desarrollo Integrado (IDE). Estos ofrecen facilidades al equipo de desarrollo debido a que permite corrección de errores comunes que se comenten a diario.[10]

EasyEclipse

El proyecto EasyEclipse tiene como principal propósito proveer una fácil instalación y uso de las distribuciones de Eclipse. Su arquitectura basada en plugins extiende las funcionalidades de la

herramienta, por ejemplo, con la adición del soporte para varios lenguajes, como pueden ser: Java, Ruby, PHP, Python o Prolog.

Para desarrollar la investigación se decidió utilizar EasyEclipse debido a que actualmente se sitúa a la vanguardia de los entornos de desarrollo para Python existentes por su alto nivel de integración con este lenguaje, completamiento de código, resaltado de sintaxis, depuración de la ejecución, funcionamiento en varias plataformas, refactorización del código, facilitando con este último el trabajo al programador. [18]

1.6.5. Framework de JavaScript

JQuery se está convirtiendo rápidamente en una herramienta que todo desarrollador de interfaces web debería de conocer. Es una biblioteca de JavaScript, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.

JQuery es software libre y de código abierto, posee un doble licenciamiento bajo la Licencia MIT y la Licencia Pública General de GNU v2, permitiendo su uso en proyectos libres y privativos; al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio.

Esta herramienta consiste en un único fichero JavaScript que contiene las funcionalidades comunes de DOM, eventos, efectos y AJAX.[19]

La característica principal de la biblioteca es que permite cambiar el contenido de una página web sin necesidad de recargarla, mediante la manipulación del árbol DOM y peticiones AJAX, aunque pose otras características como:

- Interactividad y modificaciones del árbol DOM, incluyendo soporte para CSS 1-3 y un plugin básico de XPath.
- Manipulación de la hoja de estilos CSS.
- Efectos y animaciones.
- Animaciones personalizadas.
- Soporta extensiones.[10]

1.7. Conclusiones parciales

Gracias a las definiciones y conceptos brindados en este capítulo se llegó a una mejor comprensión del proceso de replicación y sincronización de bases de datos, además de que se logró un mayor conocimiento de herramientas con funcionalidades similares a la propuesta, pero tienen un precio demasiado alto para ser utilizadas en el país.

Gracias al estudio realizado se escogieron las metodologías, lenguajes de modelado, lenguajes de programación, IDE y herramienta CASE, necesarias y afines para la realización del software.

La propuesta de desarrollo presente tiene una gran cantidad de herramientas populares en el mundo informático, además de que tienen libre patente, lo cual es un requisito importante para el desarrollo del sistema.

CAPITULO 2: CARACTERÍSTICAS DEL SISTEMA

2.1. Introducción

En este capítulo se tratará la situación problemática así como la propuesta del sistema, características, seguridad, soporte, rendimiento, requerimientos funcionales y no funcionales, para hacer posible la sincronización y replicación de bases de datos. Se definirán los casos de uso modelando el sistema propuesto a través de los actores y los casos de uso que interactúan con el sistema, así como una descripción expandida de ellos donde se especifica el rol que cumplen al ejecutar cada uno de los procesos existentes en la solución que se propone. Se especificarán los requisitos del software, del mismo modo incluye una síntesis de las dependencias y relaciones con otros software, usabilidad, portabilidad confidencialidad, ayuda y documentación en línea. Se conceptualizarán las principales entidades y sus relaciones, definiendo las necesidades y principales funcionalidades del sistema.

2.2. Propuesta del sistema

Figura 5. Propuesta del Sistema.

La solución parte de mantener lo logrado en el Organismo Central del MIC, disminuyendo los recursos humanos, materiales y asegurando la información.

La solución se compone de un dispositivo de hardware y una aplicación de software. El hardware es un dispositivo electrónico desarrollado por ICID de conexión USB con almacenamiento de tecnología Flash conectado a dos servidores de bajas prestaciones y el software es un módulo agregado a una aplicación

realizada en la UCI, llamado CID-555: Software para Intercomunicación de Redes Aisladas Físicamente, que gestiona la entrada y salida de correos en un servidor de transporte de correos MTA y controla el dispositivo de hardware. El sistema puede ser conectado y desconectado de la infraestructura tecnológica de servicios sin afectar el flujo de información.

2.3. Descripción del sistema

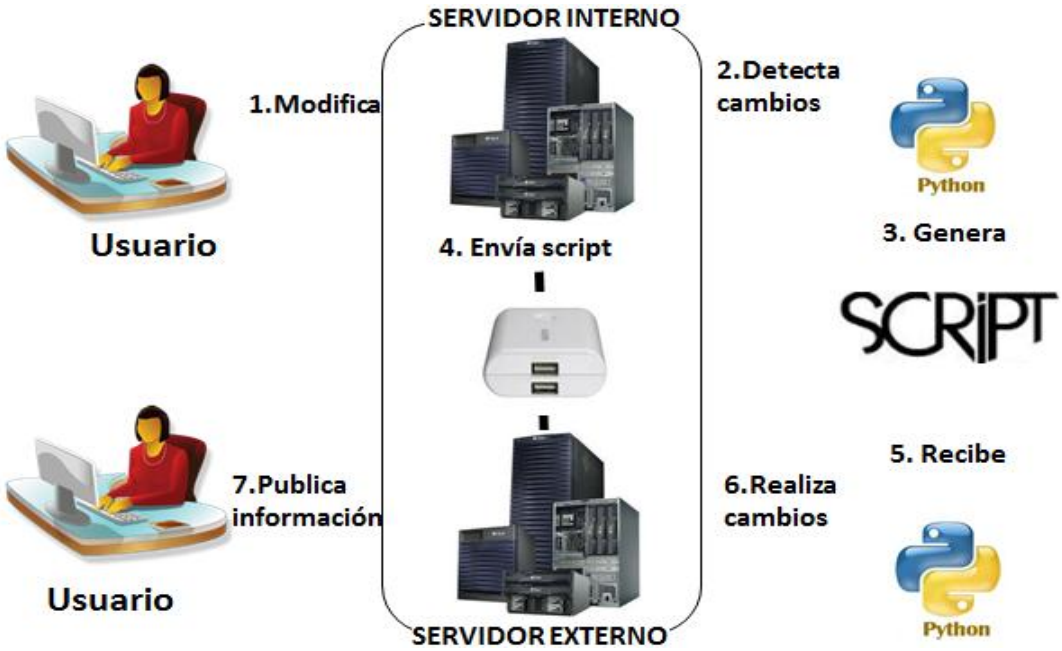


Figura 6. Descripción del Sistema.

La solución de agregar un módulo que sincronice y replique bases de datos garantiza el siguiente flujo de actividades:

- El administrador realiza cambios en la base de datos Principal.
- El Reko ubicado en el servidor interno detecta las modificaciones y genera un script con los cambios ocurridos.
- El script se guarda en la memoria 1 conectada al servidor interno mediante el dispositivo CID-555.
- El dispositivo CID-555 realiza la conmutación de las memorias, quedando conectado al servidor externo, la memoria 1 y la memoria 2 conectada al servidor interno.

- Se copia el script ubicada en la memoria 1 para la carpeta local del Reko.
- El Reko sincroniza el servidor externo mediante el script con las actualizaciones haciendo posible publicar la información con la veracidad requerida.
- El dispositivo vuelve a conmutar los dispositivos de almacenamiento quedando nuevamente la memoria 1 conectada al servidor interno y la memoria 2 conectada al servidor externo.

2.4. Modelo de negocio

El proceso inicia cuando el usuario realiza cambios en la base de datos Principal del servidor interno, el administrador, dado un tiempo determinado, siempre y cuando hayan ocurrido cambios en la base de datos principal, genera un script con los cambios realizados. Este script es guardado en un dispositivo de almacenamiento USB o memoria flash, el dispositivo realiza la conmutación de las memorias, para luego copiar el script en el servidor externo. La aplicación actualiza el servidor externo con la información que se encuentra en el script, y por último el usuario consulta la aplicación para ver los cambios ocurridos.

A continuación se muestra en la figura 7 el diagrama de procesos de negocio para el módulo Sincronización de Base de Datos.

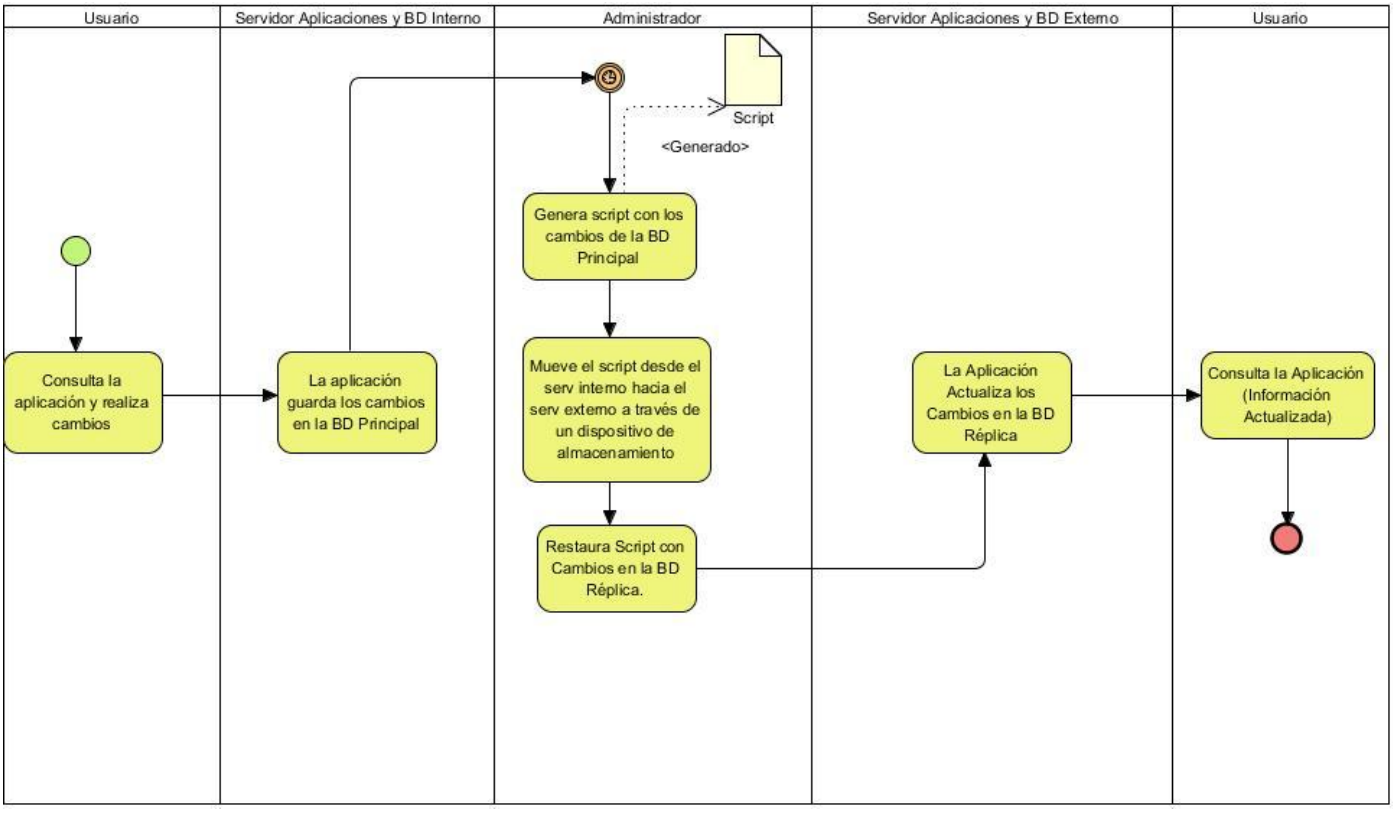


Figura 7. Diagrama de Procesos de Negocio CID-555. Módulo Sincronización de Base de Datos.

2.5. Especificación de los requisitos de software

Luego de la realización del modelo de negocio, se procede a ejecutar el proceso de captura de requisitos del sistema, para el cual se visitó en varias ocasiones el MIC.

Los requisitos funcionales establecen los comportamientos del sistema como son: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que muestran cómo los casos de uso serán llevados a la práctica.

2.5.1. Requerimientos funcionales

Se describieron 21 requerimientos funcionales:

RF1. Crear carpetas en las memorias flash y en la ubicación local de la pc.

RF2. Copiar script en las memorias flash.

RF3. Iniciar y detener el sistema.

RF4. Generar Logs.

RF5. Encriptar la información.

RF6. Decriptar la información

RF7. Aplicar suma de verificación.

RF8. Empaquetar mensajes.

RF9. Desempaquetar mensajes.

RF10. Chequear capacidad de la memoria flash.

RF11. Mostrar el log del registro de actividad en el sistema.

RF12. Alertar eventos inesperados.

RF13. Establecer tiempo de espera del ciclo de iteraciones.

RF14. Establecer tiempo de espera de intercambio de memorias flash.

RF15. Configurar en el Reko la sincronización de base datos.

RF16. Generar script a partir de los cambios en la base de datos principal.

RF17. Empaquetar script y copiarlo en el dispositivo de almacenamiento.

RF18. Intercambiar script.

RF19. Desempaquetar script copiado del dispositivo de almacenamiento.

RF20. Importar en la base de datos réplica.

RF21. Mostrar la fecha de la última sincronización.

Descripción de los requerimientos de software.

RF1: Se crea la carpeta en el dispositivo de almacenamiento donde se va a guardar el script con los cambios ejecutados en la Base de Datos Principal si el usuario no la ha creado previamente.
RF2: Se copia el script con los cambios ejecutados en la Base de Datos Principal en el dispositivo de almacenamiento.
RF3: Se detiene el funcionamiento del sistema hasta que el usuario vuelva a iniciarlo.

RF4: Se crean los logs como una forma de mantener al usuario informado de lo que va ocurriendo en el sistema.

RF5: Se encripta el script utilizando el método de encriptación AES o Algoritmo de Rijndael y una clave conocida en ambos servidores evitando así el conocimiento de la información contenida en éste, imposibilitando su utilización de caer en manos con malas intenciones. [20]

RF6: Se desencripta utilizando la clave para actualizar la Base de Datos Réplica.

RF7: Se aplica la suma de verificación al script obtenido del dispositivo de almacenamiento para comprobar mediante una clave si es el mismo que partió desde la Base de Datos Principal.

RF8: Se crea un archivo compactado que contiene los script, y durante el proceso se les hace el proceso de encriptamiento.

RF9: Se desempaqueta el archivo para desencriptarlo y hacer las actualizaciones pertinentes.

RF10: Se comprueba que el dispositivo de almacenamiento tenga espacio suficiente para guardar el script.

RF11: Se guarda el registro de actividad en un Log en el sistema de ficheros de manera quede persistente para análisis forense. Además se muestra en tiempo real en la interfaz de la aplicación.

RF12: Muestra al usuario si han ocurrido eventos inesperados, como caída de conexión con el dispositivo, fallo de flujo eléctrico, etc.

RF13: El administrador establece un tiempo de espera para las iteraciones tales como el intercambio de las memorias.

RF14: Se establece un tiempo para que el administrador haga el cambio de la memoria de un servidor al otro.

RF15: Se configura en el Reko la sincronización de las Bases de Datos definiendo parámetros como son, la carpeta en la que se va a guardar el script con los cambios generados en la Base de Datos Principal, se establece el gestor de Base de Datos con que se va a trabajar y se establecen las Bases de Datos a sincronizar.

RF16: El Reko genera un script a partir de los datos introducidos por el usuario en la Base de Datos Principal.

RF17: Luego de empaquetado el script se copia en el dispositivo de almacenamiento.

RF18: Se copia en la Base de Datos Réplica el script generado en la Base de Datos Principal.

RF19: Se desempaqueta el script obtenido desde el dispositivo de almacenamiento.

RF20: Se actualiza la Base de Datos Réplica con la información obtenida en el script.
RF21: Se muestra en la carpeta donde se guardan temporalmente los script, el último script de actualización con la fecha en que fue guardado.

2.5.2. Requerimientos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener, es decir, son las características que hacen al producto atractivo, usable y confiable. Normalmente, están vinculados a los requerimientos funcionales, o sea, una vez que se conozca lo que el sistema debe hacer, se puede determinar cómo ha de comportarse, las cualidades que debe tener o cuán rápido o grande debe ser.

RNF1. Apariencia o interfaz externa

- El sistema informático contará con un diseño de interfaz sencillo, agradable, sugerente e intuitivo, de fácil entendimiento. Las páginas de la aplicación estarán poco cargadas, sólo contendrá la información requerida para el usuario.

RNF2. Usabilidad

- La aplicación podrá ser manejada por cualquier usuario con conocimientos básicos del tema. La interfaz y los mensajes para interactuar con el usuario, así como los mensajes de error, serán en el lenguaje español. Los mensajes de error deben ser lo suficientemente informativos para dar a conocer la severidad del error. Los elementos de navegación deben ser orientados al usuario.

RNF3. Soporte

- El sistema debe contar con un manual de usuario que permita un mayor uso de sus funcionalidades y una mayor experiencia en el trabajo con la aplicación.
- El sistema debe contar con un manual de instalación el cual podrá ser consultado para los elementos necesarios en el despliegue del sistema (configuraciones, dependencias y elementos a instalar).

RNF4. Seguridad

- El sistema solo debe permitir interactuar con la aplicación a los usuarios que tengan acceso según el rol definido.
- La contraseña de los usuarios del sistema será encriptada para que no pueda ser leída en el fichero donde será guardada.
- El sistema solo interactuará con la memoria flash destinada para esta función, previendo de esta forma la suplantación de la memoria. Esto se garantiza teniendo la memoria un id definido e irrepetible.
- La información de las memorias flash estará encriptada, previendo de esta forma el robo de información.
- A los paquetes recibidos en cada iteración se le realiza una suma de verificación para comprobar la integridad de la información.

RNF5. Legales

- El sistema y la documentación del mismo, pertenecen a la Universidad de las Ciencias Informáticas.
- El sistema es desarrollado con herramientas libres lo que permite independencia tecnológica.

RNF6. Confiabilidad

- El sistema debe ser capaz de recuperarse ante la ocurrencia de un fallo y alertar al personal encargado de la administración del mismo, así como proteger la información y el contenido de los dispositivos de almacenamiento.

RNF7. Software

- El Servidor de Aplicaciones Web es Apache con soporte para SSL.
- El lenguaje de programación del sistema es Python.
- Los lenguajes de programación de la interfaz son PHP y Jquery.
- Las computadoras clientes deben tener instalado un navegador web Mozilla Firefox.
- Se requiere la instalación de un Sistema Operativo GNU/Linux, se recomienda Ubuntu.

RNF8. Hardware

- Se requieren 2 servidores que tengan como mínimo 1 GB de memoria RAM, procesador Pentium IV o superior y 100 GB de disco duro disponibles.[10]

2.6. Conclusiones parciales

En el capítulo desarrollado se ha efectuado un análisis ingenieril de las características del sistema que se presenta, lo cual contribuye a la documentación y correcta implementación de la aplicación. Para un mayor entendimiento del sistema se diseñó el diagrama del negocio del sistema con los pasos principales a seguir de la aplicación. Se realizó la definición de los requisitos, tanto funcionales, como no funcionales. En este orden, además del cumplimiento de los requisitos expresados, es posible el comienzo de la construcción del sistema que se presenta.

CAPITULO 3: DISEÑO DEL SISTEMA

3.1. Introducción

El presente capítulo está enfocado en el diseño del sistema mediante la metodología RUP. Determinados los requerimientos funcionales, se definirán los casos de uso del sistema, entrada fundamental para el diseño. Se procede a analizar si es posible dar una solución que satisfaga los requisitos significativos de la arquitectura.

Teniendo en cuenta la propuesta de solución descrita en el capítulo anterior, es necesario definir cómo se desarrollará el sistema.

3.2. Definición de los Casos de Uso

Un caso de uso es un escenario que describe como el software va a ser usado en una determinada situación. Ingeniería del software. [21]

3.2.1. Diagrama de Casos de Uso del Sistema

El diagrama de casos de uso del sistema se encarga de representar mediante gráficas los procesos y su interacción con los actores. Los casos de uso son artefactos que describen, mediante acciones y reacciones, el comportamiento que tendrá el sistema desde el punto de vista del usuario.

Establece un acuerdo entre los clientes y los desarrolladores en cuanto a las condiciones y posibilidades que debe cumplir el sistema.[10]

En el diagrama se representan 3 casos de uso determinados después de haber identificado los requisitos del sistema, estos son:

Casos de uso a realizar para el Módulo de Sincronización de Base de Datos.

1. Mostrar sucesos del sistema.
2. Configurar módulo de sincronización de Base de Datos.
3. Configurar el Reko para la Sincronización de BD.

A continuación se muestra en la figura 8 el diagrama de casos de uso para el módulo Sincronización de Base de Datos.

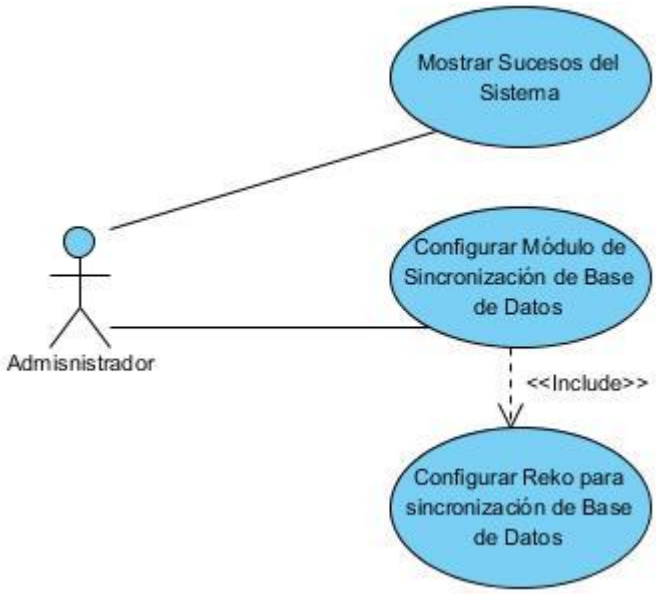


Figura 8. Diagrama de Caso de Uso CID-555. Módulo Sincronización de Base de Datos.

Descripción de los Casos de Uso del Sistema

La descripción de los casos de uso del sistema se realiza para lograr un mayor entendimiento de las funcionalidades asociadas a estos. Se hace un mayor acercamiento a las acciones y reacciones del actor, hacia el sistema y las respuestas que este último genera. Dicha descripción debe ser breve y precisa para lograr su rápido entendimiento. (Ver descripción en los anexos).

Matriz de trazabilidad

Requisitos\Casos de Uso	CU1. Mostrar sucesos del sistema.	CU2. Configurar módulo de sincronización de Base de Datos.	CU3. Configurar el REKO para la Sincronización de BD.
RF1. Crear carpetas en las memorias flash y en la ubicación local de la pc.		X	X
RF2. Copiar script en las memorias flash.		X	X
RF4. Generar Logs.	X		
RF10. Chequear capacidad de la	X		

memoria flash.			
RF11. Mostrar el log del registro de actividad en el sistema.	X		
RF14. Establecer tiempo de espera de intercambio de memorias flash.		X	X
RF15. Configurar en el Reko la sincronización de base datos.		X	X
RF20. Importar en la base de datos réplica.			X
RF21. Mostrar la fecha de la última sincronización.	X		

Tabla 1. Matriz de Trazabilidad.

Los demás requisitos funcionales se relacionan con los casos de uso definidos en la Pasarela de Correos CID-555.

3.3. Arquitectura del Sistema

El diseño del software “Software para la Interconexión de Redes Aisladas. Módulo Sincronización de Base de Datos” está dividido en dos partes: front-end y back-end.

Front-end

Es la parte del software que interactúa con el usuario final, es decir, la interfaz de administración web. Para el diseño de la propuesta de solución, se utiliza el patrón arquitectónico Modelo - Vista- Controlador, con lo cual se separan los datos de la aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

La siguiente figura representa el patrón arquitectónico: Modelo - Vista– Controlador aplicado para el módulo Sincronización de Base de Datos.

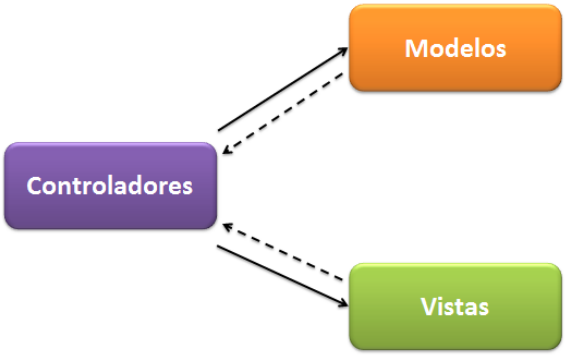


Figura 9. Diagrama de la arquitectura front-end. CID-555: Software para la Interconexión de Redes Aisladas Físicamente. Módulo Sincronización de Base de Datos.

Detalles de los componentes del diagrama:

- **Controladores:** Es donde se encuentran las clases que atienden, responden y enrutan las acciones solicitadas por el usuario, constituyen además todo lo referente a verificación de entrada de datos y cómo se presentará la información en las vistas.
- **Vistas:** Es donde se encuentran los archivos que permite la interacción entre el usuario y el sistema, para finalmente retornar una respuesta.
- **Modelos:** Es donde se encuentran las clases que procesan la lógica del negocio, éstos interactúan con la Base de Datos y con el Back-end.

Back-end

Es la parte del software que procesa la entrada desde el front-end. Para el diseño de la propuesta de solución del mismo, se utiliza el patrón arquitectónico: 3-Capas con dos aspectos añadidos denominados Seguridad y Dominio. En este patrón arquitectónico el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño; un ejemplo básico de esto consiste en separar la capa de datos de la capa de presentación al usuario. Las capas puedan ser manejadas e implementadas de forma independiente, y poseerán responsabilidades específicas que no dependan del funcionamiento de las otras o al menos que su dependencia sea mínima. Este aspecto constituye una ventaja considerable pues proporciona una amplia reutilización de las clases implementadas al hacer abstracciones de las distintas funcionalidades o responsabilidades del sistema agrupándolas en capas.

En la figura 10 se representa la distribución de los componentes del sistema para el módulo Sincronización de Base de Datos.

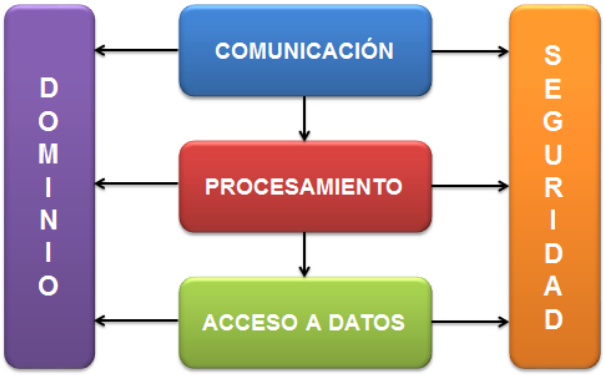


Figura 10. Diagrama de la arquitectura back-end. CID-555: Software para la Interconexión de Redes Aisladas Físicamente. Módulo Sincronización de Base de Datos.

Detalles de los componentes del diagrama:

- **Comunicación:** Capa que permite la comunicación entre la interfaz web y la aplicación de control del proceso de intercambio de mensajes.
- **Procesamiento:** Es la capa encargada de iniciar y manejar todo el proceso del sistema, es la que realiza los cálculos de procesamiento.
- **Acceso a Datos:** Capa encargada de interactuar con la Base de Datos.
- **Seguridad:** Es el aspecto que contiene todos los elementos necesarios para la implementación de la seguridad del sistema.
- **Dominio:** Es el aspecto que brinda la posibilidad del manejo de los objetos del dominio del sistema, de manera tal que todas las capas del sistema puedan acceder a las funcionalidades que esta capa brinda. [10]

3.4. Patrones de Diseño Utilizados

Los patrones de diseño son una solución reutilizable de problemas recurrentes que ocurren durante el desarrollo del software.

Para el diseño de la aplicación se hizo uso de los Patrones Generales de Software para Asignar Responsabilidades (GRASP).

Los patrones de GRASP utilizados fueron:

- **Alta cohesión:** Se aplica en la mayoría las clases del diseño, ya que en cada una solo se implementan las funcionalidades que le corresponden.
- **Bajo acoplamiento:** Ya que cada clase se comunica con un número relativamente pequeño de clases.
- **Creador:** Las clases que tienen la responsabilidad de crear objetos contienen toda la infromación necesaria para construir los mismos.
- **Experto:** Se mantiene el encapsulamiento, los objetos utilizan su propia información para llevar a cabo sus tareas. Se distribuye el comportamiento entre las clases que contienen la información requerida. Son más fáciles de entender y mantener.

Se utilizaron también patrones del Gang of Four (GoF); o Pandilla de los Cuatro:

Creacionales

- **Singleton(Instancia única):** Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia.[10]

3.5. Diagrama de Clases de Diseño

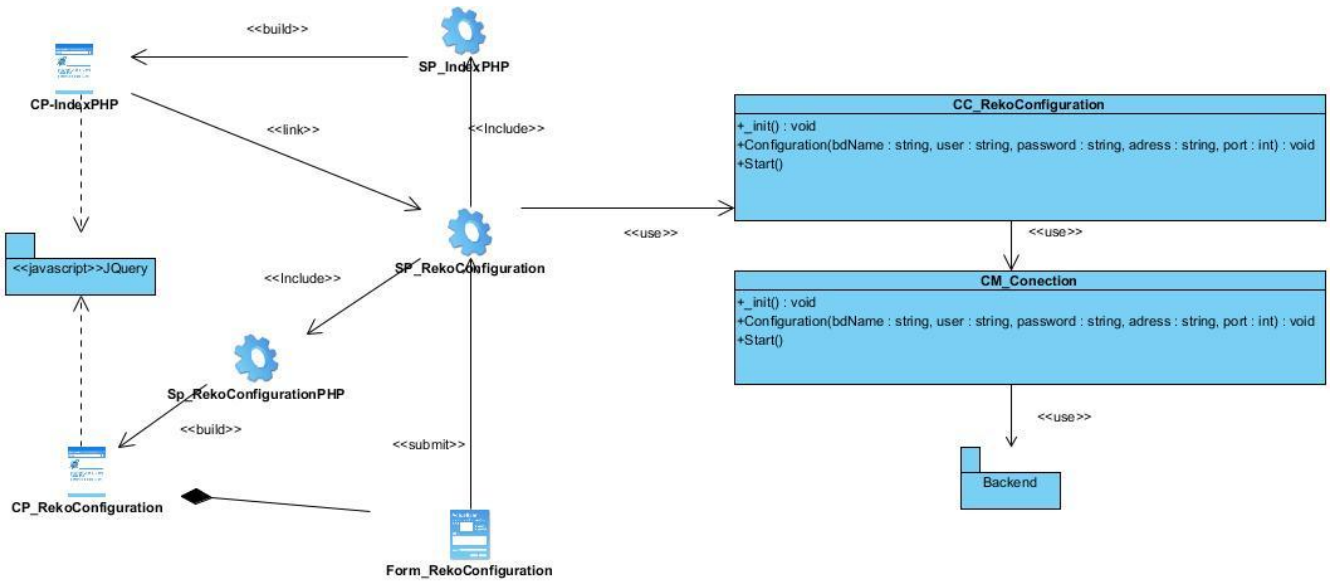


Figura 11. Diagrama de Clase de Diseño Configurar Módulo para Sincronización de Base de Datos.

Descripción:

La server page principal SP_RekoConfiguration es la que se encargar de manejar todos las actividades, en este caso manda la server page SP_IndexPHP a construir la client page CP_IndexPHP con la que interactuará el administrador. Este selecciona la opción de configurar la aplicación Reko y la página principal hace un link para enviar este pedido a la server page SP_RekoConfiguration. Esta última manda a la server page SP_RekoConfigurationPHP a construir la client page CP_RekoConfiguration la cual está compuesta por un formulario en el cual se archivará la información obtenida de la entrada de datos del administrador. El formulario envía la información recogida a la server page SP_RekoConfigurationPHP, el cual envía los datos a las clase controladora, quien hace la conexión con la clase modelo con los datos obtenidos y los envía al back-end del sistema, que es el que se encarga de manejar esta información.

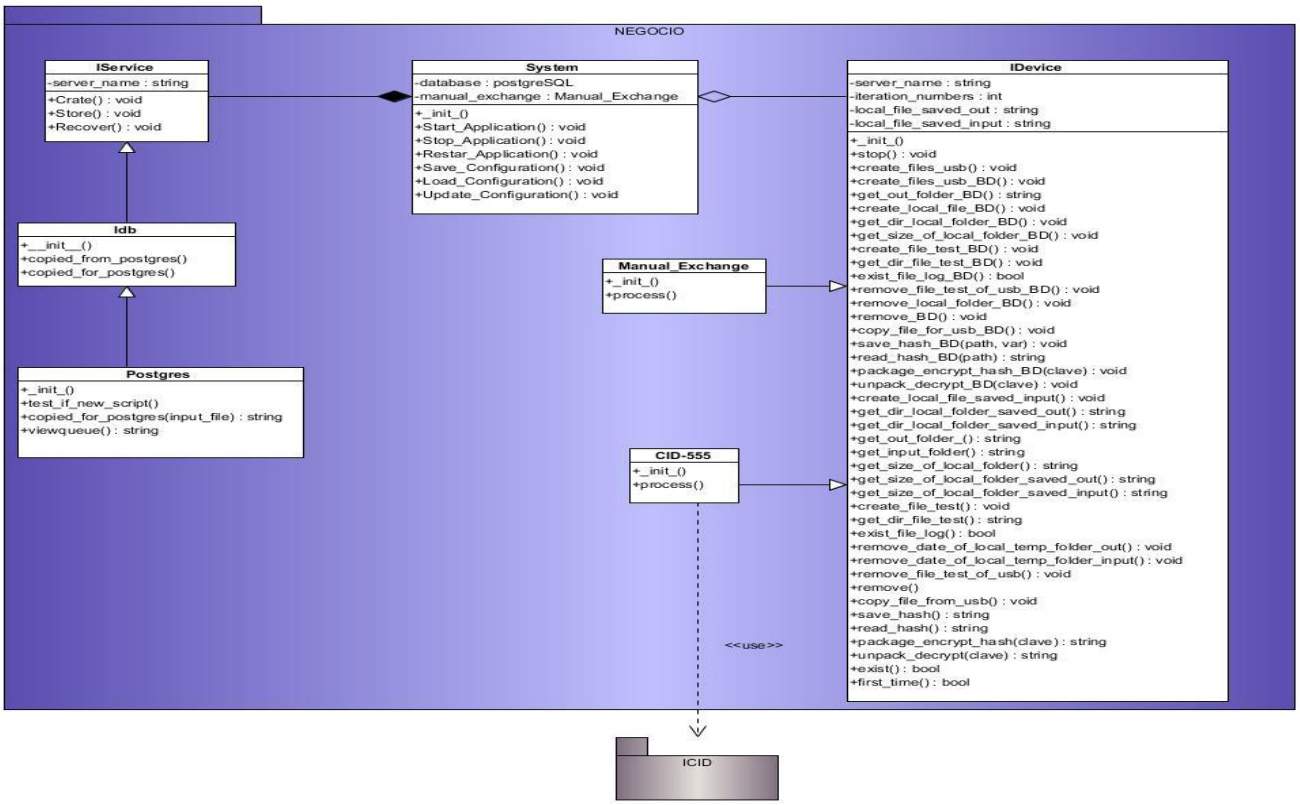


Figura 12. Diagrama de Clase de Diseño Capa de Negocio.

Descripción:

En el Diagrama de Clase de Diseño Capa de Negocio del módulo a desarrollar se presentan las relaciones entre las principales clases existentes.

La clase Postgres que contiene los métodos para saber si hay nuevo script y de copiar script de la bandeja local de Reko, hereda de la clase Idb que a su vez tiene los métodos para copiar para el postgres y copiar desde el postgres.

La clase Idb hereda de la clase IService la cual se encarga de crear, recuperar y guardar la base de datos.

La clase System está compuesta por la IService y sus herencias, dicha clase es la que se encarga de manejar las diferentes etapas del sistema, como son, detenido, inicializado, actualizado, reiniciado, entre otros.

La clase IDevice se agrega a la clase System, esta clase maneja todo lo referente a los dispositivos de almacenamiento, ya sea, la creación de las carpetas, obtener la capacidad de almacenamiento, obtener las direcciones de las carpetas contenidas dentro de ellos, etc. Maneja también lo relacionado con la información contenida dentro de los dispositivos de almacenamiento, se encarga de empaquetar los scripts, aplicarles el encriptamiento, el hash, hacer la suma de verificación, borrar los scripts, entre otras.

La clase Manual_Exchange que hereda de la clase IDevice contiene el método Process, cuya función es manejar todo lo que tiene que ver con la conmutación de las memorias a través del dispositivo, ya sea de forma manual o automática.

Y la clase CID-555 que contiene el método process es la encargada de manejar todo el sistema, de hacer la llamada a los métodos contenidos en estos y hacer uso de las clases contenidas en este y controla el dispositivo ICID.

3.6. Diagrama de Secuencia

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso. Para manejar la secuencia de acciones de un caso de uso es más factible utilizar este tipo de diagrama para concentrar la atención en encontrar las secuencias de interacciones ordenadas por el tiempo y de forma más detallada.

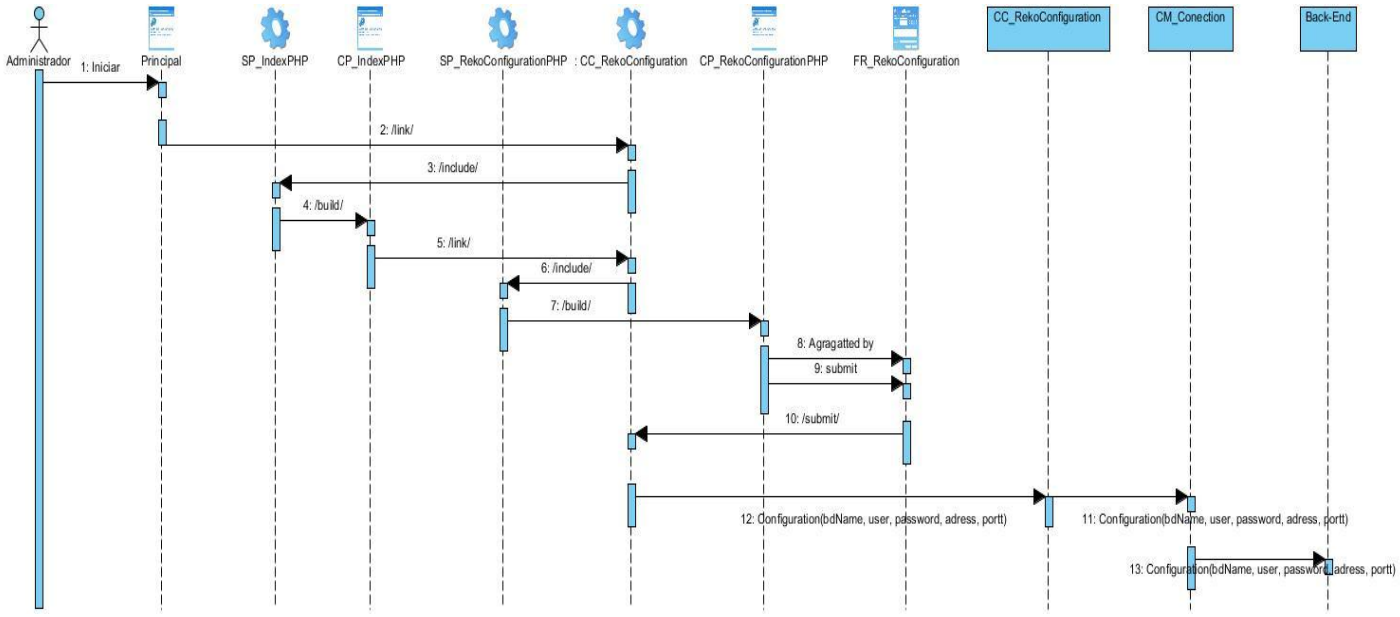


Figura 13. Diagrama de Secuencia. CU: Configurar Sistema.

Descripción:

En el Diagrama de Clase de Secuencia para el caso de uso Configurar Sistema se muestra el flujo de actividades a seguir cuando el administrador hace la configuración. Primeramente el administrador mediante la client page Principal inicia el sistema. Esta hace un link a la server page principal SP_RekoConfigurationPHP, quien manda a que esta construye la client page CP_IndexPHP. En esta client page el administrador escoge la opción de configurar Reko, hace un link a la server page principal quien manda a la server page SP_RekoConfigurationPHP a construir la client page CP_RekoConfigurationPHP para que el administrador entre los datos de la configuración, luego de obtenidos los datos a través del formulario FR_RekoConfiguration este los envía a la server page principal SP_RekoConfigurationPHP quien a su vez los envía para la clase controladora CC_RekoConfiguration, la cual inicia la conexión con la clase modelo CM_Conection para guardar estos datos, para luego ser enviados al back-end de la aplicación para que este maneje esa información.

3.7. Diseño de la Base de Datos

Una base de datos correctamente diseñada permite obtener acceso a información exacta y actualizada. Debido a que un diseño correcto es esencial para lograr los objetivos fijados para la base de datos, se siguen una serie de pasos para la correcta construcción de esta. Lo primero es definir su estructura para

poder almacenar la información y posteriormente recuperarla, luego dividirla en tablas basadas en temas para reducir los datos redundantes y lo más importante es que la información sea correcta y completa.
A continuación se muestra el diagrama de clases persistentes y el modelo entidad-relación de la misma.

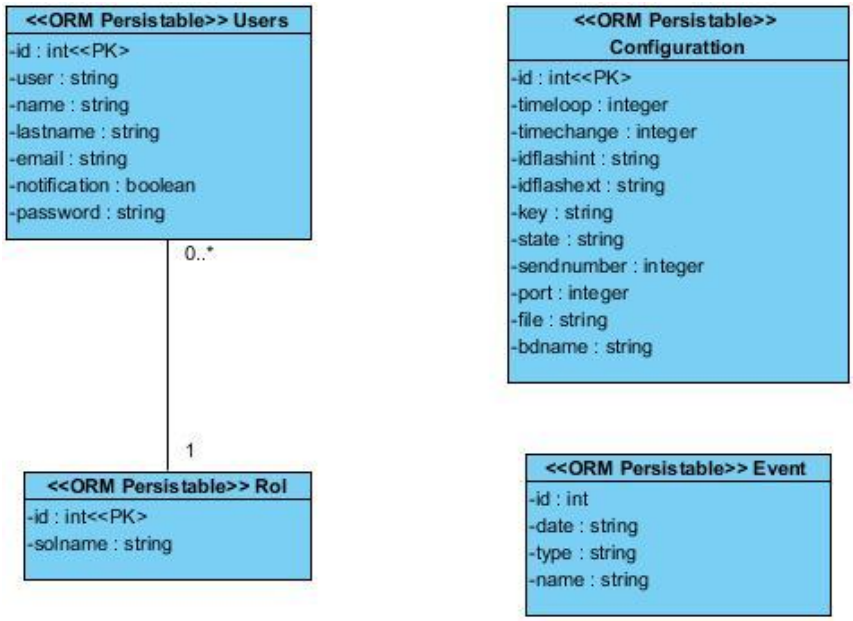


Figura 14. Diagrama de Clases Persistentes.

Descripción:
En el diagrama de clases persistentes se establecen los valores que se archivarán en la base de datos SQLITE. Estas son los usuarios, de los cuales se conocerá además el rol que tienen, para poder definir los permisos que tienen al autenticarse en el sistema.
Además se encuentra también la configuración con la que el sistema funcionará, como por ejemplo, el tiempo de conmutación de las memorias, los id de las memorias con las que trabajará el dispositivo de conmutación, etc.
Y por último los eventos que suceden en el sistema como método de informar al administrador de lo que va ocurriendo en el sistema, los mensajes se componen de la fecha en la que ocurrió la acción, el tipo de acción y una descripción de la misma.

3.8. Conclusiones Parciales

En este capítulo se ha presentado un estudio relacionado con el diseño de la aplicación que sirve de base para la implementación del sistema. El diseño de software materializó los requerimientos del cliente. Se ha

logrado modelar todos los procesos que han sido objeto de estudio durante el transcurso de la investigación, lo que proporciona una idea completa de lo que realmente es el software que se presenta, permitiendo describir todos los aspectos del futuro sistema. Los diagramas y especificaciones de diseño que se proponen constituyen una guía que puede ser fácilmente comprendida por los desarrolladores con el objetivo de implementar la aplicación que se ha diseñado.

CAPITULO 4: IMPLEMENTACIÓN Y PRUEBA

4.1. Introducción

En este capítulo se profundiza el tema de la implementación del sistema, donde se pueden ver los subsistemas en un modelo de implementación así como sus componentes. Se tiene en cuenta el diagrama de despliegue para tener una visión de los componentes del sistema, además de las capas que lo conforman. Se implementan las clases y sus relaciones contando con los lenguajes de programación.

4.2. Generalidades de la implementación

En esta fase se lleva a cabo la implementación del sistema siguiendo la arquitectura y los patrones de diseño escogidos.

El Modelo de Implementación está compuesto por un conjunto de componentes. Entre los componentes podemos encontrar datos, archivos, ejecutables, código fuente y los directorios. Fundamentalmente, se describe la relación que existe desde los paquetes y clases del modelo de diseño a subsistemas y componentes físicos.

Este artefacto describe cómo se implementan los componentes, congregándolos en subsistemas organizados en capas y jerarquías, y señala las dependencias entre éstos.[22]

4.3. Diagrama de despliegue

El diagrama de despliegue se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes. Describen la topología del sistema, la estructura de los elementos de hardware y el software que ejecuta cada uno de ellos. Los diagramas de despliegue representan a los nodos y sus relaciones.[23]

En la figura12 se muestra el modelo de despliegue para el sistema propuesto:

Figura 15. Diagrama de Despliegue.

Descripción:

El sistema cuenta con un servidor de aplicación conectado a través del protocolo TCP/Ip a un servidor de base de datos en cada subred y comunicados entre ellos por el dispositivo de conmutación de las memorias.

4.4. Diagrama de componentes

Un diagrama de componentes permite visualizar con más facilidad la estructura general del sistema y el comportamiento del servicio que estos componentes proporcionan y utilizan a través de las interfaces.

Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables, o paquetes.

Los diagramas de Componentes prevalecen en el campo de la arquitectura de software pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema.[24]

A continuación se muestran los diagramas de componentes de la aplicación desarrollada:

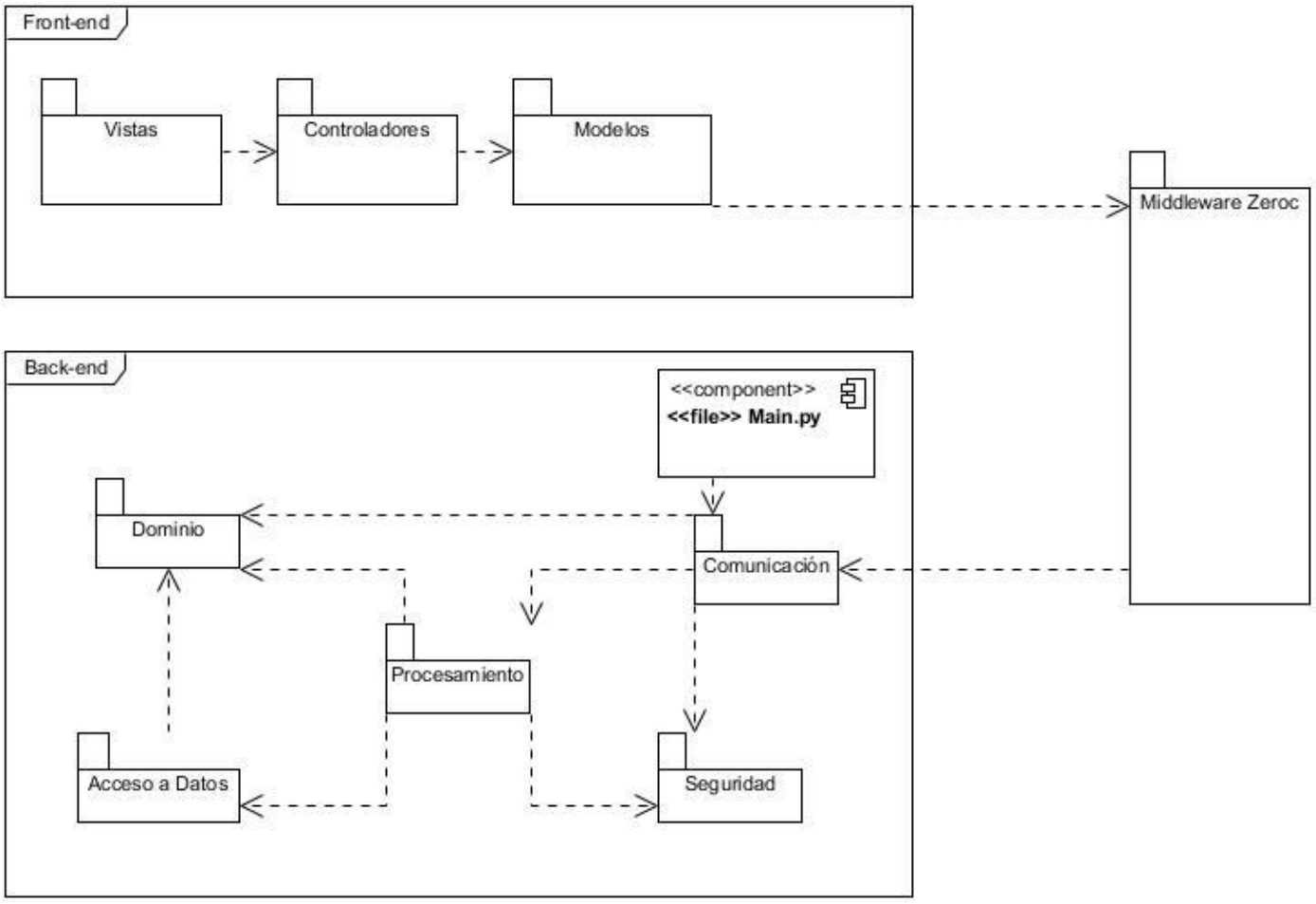


Figura 16. Diagrama de componentes de todo el sistema.

Descripción:

En el paquete de componentes del front-end se encuentran las capas del patrón arquitectónico utilizado, el patrón Modelo-Vista-Controlador, compuesto por las capas Vistas, Controladores y Modelos, mientras que el paquete de componentes del back-end se encuentra distribuido utilizando el patrón arquitectónico 3-Capas con dos aspectos incluidos, las capas de este patrón arquitectónico Comunicación, Procesamiento y Acceso a Datos interactúan con los aspectos denominado Dominio y Seguridad. El front-end y el back-end se comunican a través del paquete de Middleware Zeroc y de la capa de Comunicación del back-end.

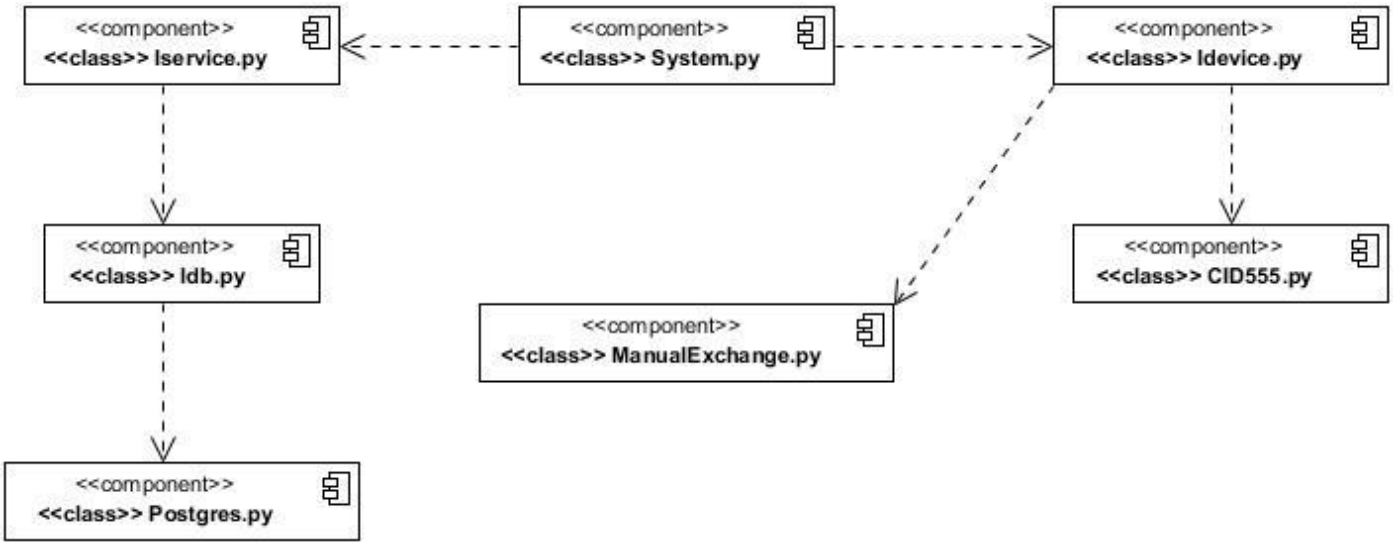


Figura 17. Diagrama de componentes. Capa de Negocio.

Descripción:

El diagrama de componentes de la capa de negocio evidencia la interacción de los componentes del negocio con el objetivo de que el sistema funcione de forma correcta.

A partir del componente que contiene la clase System.py, la cual se encarga de manejar las conexiones con la base de datos, además de todo lo relacionado con la configuración del sistema, ya sea configurarlo, inicializarlo, detenerlo, cambiar el id de las memorias, entre otras, surgen las relaciones con el componente compuesto por la clase Iservice.py y el componente integrado por la clase Idevice.py. Siguiendo el flujo a partir del primero se tiene la relación con el componente que tiene la clase Idb.py, y esta a su vez se relaciona con el componente que contiene la clase Postgres.py, con estos componentes se garantiza a través de la clase Iservice.py comprobar si hay un script nuevo, con la clase Idb.py se manda a hacer la copia el script tanto para la carpeta de salida del postgres actualizado como a la carpeta de entrada del postgres a sincronizar, y la clase Postgres.py es la que hace estas acciones. Siguiendo la segunda relación el componente con la clase Idevice.py se relaciona de forma simultánea con los componentes que contienen las clases ManualExchange.py y CID555.py, la primera se encarga básicamente de realizar la conmutación de las memorias en el dispositivo de forma manual, con todo el flujo de actividades que esta operación trae consigo, dígase comprobar que la memoria esté conectada, que su id sea el esperado, copiar los archivos, hacerles el encriptamiento y empaquetamiento, entre otras

operaciones, y la clase CID555 es la que se encarga de manejar todo el flujo de operaciones para que el sistema funcione correctamente.

4.5. Caso de pruebas

Los casos de prueba o *test case* son un conjunto de condiciones en las que se introducen datos o variables, con el objetivo de obtener diversos resultados bajo los cuáles se determinará si el requisito de una aplicación es parcial o completamente satisfactorio. Cuando el caso de prueba detecta errores que no han sido encontrados hasta el momento, entonces, se puede decir que es un caso de prueba aceptable.

4.6. Métodos de prueba

4.6.1. Prueba de Caja Blanca

Las pruebas de caja blanca del software se basan en el minucioso examen de los detalles procedimentales. Se comprueban los caminos lógicos del software proponiendo casos de prueba que ejerciten conjuntos específicos de condiciones y/o bucles, por lo que su diseño está fuertemente ligado al código fuente. Es un metodo de diseño de casos de prueba que usa la estructura de control del diseño procedimental para obtener los casos de prueba.

Mediante los métodos de prueba de caja blanca, el ingeniero del software puede obtener casos de prueba que:

1- Garanticen que se ejercita por lo menos una vez todos los caminos independientes de cada módulo.

2- Ejerciten todas las decisiones lógicas en sus vertientes verdadera y falsa.

3- Se ejecuten todos los bucles en sus límites y con sus límites operacionales.

4- Y ejerciten las estructuras internas d datos para asegurar su validez. [21]

El probador escoge distintos valores de entrada para examinar cada uno de los posibles flujos de ejecución del programa y cerciorarse de que se devuelven los valores de salida adecuados. Gracias a estas pruebas son chequeados los caminos lógicos del software y se puede comprobar si el estado final del sistema es el esperado.

Las pruebas de caja blanca no solo evalúan el comportamiento del usuario con la interfaz, sino que también busca errores en el código fuente.

A continuación se muestra la figura 18 donde se refleja la el flujo de acción de unos de los métodos más importantes del sistema.

```
def copied_for_postgres(self, input_file):
    event = Event()
    copied = 1
    '''Moviendo el script de la carpeta  local de entrada para el REKO'''
    if os.path.isdir('/opt/BD') is False:
        subprocess.Popen("mkdir /opt/BD",\ shell=True)
        time.sleep(1)
        good = subprocess.call("cp"+input_file+"/* /opt/BD/",\
                stdout=subprocess.PIPE, shell=True)
    if good == 0:
        copied = 0
        '''Establecer permisos de lectura al  nuevo scripts.'''
        good = subprocess.call("chmod -R 0700 /opt/BD",\
                stdout=subprocess.PIPE, shell=True)
        if good == 0:
            copied = 0
            event.save_event_log_data_base(EnumInformationOnlyLogs().script_sent,
'importar script')
            event.save_event_log_file(EnumInformationOnlyLogs().script_sent, 'importar
script')
        else:
            copied = 1
            event.save_event_log_data_base(EnumError().script_not_sent, 'error')
            event.save_event_log_file(EnumError().script_not_sent, 'error')
    else:
        copied = 1
        event.save_event_log_data_base(EnumError().script_not_sent, 'error')
        event.save_event_log_file(EnumError().script_not_sent, 'error')
    return copied
```

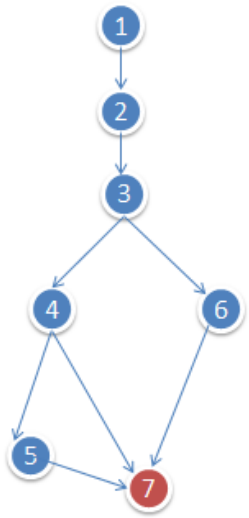


Figura 18. Grafo de Flujo.

Cálculo de la complejidad ciclomática

La complejidad ciclomática es una métrica del software que proporciona una medición cuantitativa de la complejidad lógica de un programa. Es muy útil para predecir los módulos que son más propensos a error.[21]

Es una medida que proporciona una idea de la complejidad lógica de un programa.

- La complejidad ciclomática coincide con el número de regiones del grafo de flujo
- La complejidad ciclomática, V(G), de un grafo de flujo G, se define como:

$$V(G) = Aristas - Nodos + 2$$

- La complejidad ciclomática, V(G), de un grafo de flujo G, también se define como:

$$V(G) = Nodos\ de\ predicado + 1$$

A partir del grafo de flujo de la figura 15, la complejidad ciclomática sería:

1. El número de regiones (cantidad de regiones cerradas) del grafo de flujo más una unidad (región exterior) coincide con la complejidad ciclomática. Por lo que:
   V(G) = 2 + 1 = 3
2. V(G) = (A – N) + 2 donde "**A**" es el número de aristas del grafo de flujo y "**N**" es el número de nodos del mismo.
   V(G) = 8 – 7 + 2 = **3**
3. V(G) = P + 1 donde "P" es el número de nodos predicado (nodos con más de una arista saliente) contenidos en el grafo. Los nodos 3 y 4 son nodos predicados.
   V(G) = 2 + 1 = **3**

Descripción del método:

- Se comprueba que exista la ruta de carpetas donde se va a copiar el script.
- Si no esta la dirección, se procede a crearla.
- Se le asigna a la variable **good** el resultado del método de darle los permisos de lectura al script y a la variable **copied** se le asigna **0**.
- Si **good** tiene valor **0** se procede a realizar la importación del script y se muestran los mensajes de "Importar scrip" y se le asigna a **copied** el valor de **0**.
- De lo contrario, se muestra el mensaje "Error" y se le asigna a **copied** el valor de **1.**
- Se retorna el valor de **copied**.

La variable **os.path.isdir('/opt/BD')** Devuelve verdadero si esa dirección existe.

4.6.2. Prueba de Caja Negra

Las pruebas de caja negra se centran en los requisitos funcionales del software. O sea, permiten al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. Intenta encontrar errores de las siguientes categorías:

1- Funciones incorrectas o ausentes.

2- Errores de interfaz.

3- Errores en estructuras de datos o en acceso a bases de datos externas.

4- Errores de rendimiento.

5- Errores de inicialización y de terminación. [21]

En teoría de sistemas y física, se denomina caja negra a aquel elemento que es estudiado desde el punto de vista de las entradas que recibe y las salidas o respuestas que produce, sin tener en cuenta su funcionamiento interno. En otras palabras, de una caja negra nos interesará su forma de interactuar con el medio que le rodea (en ocasiones, otros elementos que también podrían ser cajas negras) entendiendo qué es lo que hace, pero sin dar importancia a cómo lo hace. Por tanto, de una caja negra deben estar muy bien definidas sus entradas y salidas, es decir, su interfaz; en cambio, no se precisa definir ni conocer los detalles internos de su funcionamiento.

En el anexo 4 se muestra un ejemplo de caja negra realizada a la aplicación del Reko.

4.7. Conclusiones parciales

Durante el desarrollo de este capítulo se mostraron las principales relaciones de los componentes del sistema, los objetos en términos de componentes y la organización de las diferentes clases.

Se utilizó el Diagrama de Despliegue para comprender de forma más detallada y clara toda la distribución de la aplicación además de las relaciones entre los componentes del sistema.

Se estableció como método de comprobación las pruebas de caja blanca para comprobar que el sistema cumple con los requerimientos establecidos.

Se da por concluida así la implementación de la aplicación dando por cumplido el objetivo general trazado al inicio del desarrollo del sistema.

CONCLUSIONES

Debido a que actualmente el país se está adentrando cada días más en lo que es el campo de publicación de datos en Internet mediante los sitios web, es de suma importancia que esta información sea genuina. Ya que en el mundo existen personas mal intencionadas, capaces de acceder a la información a publicar y cambiarla por datos incorrectos, se considera que la adición del Módulo de Sincronización de Bases de Datos al Software para la Interconexión de Redes Aisladas, sería una medida más que adecuada para evitar estos ataques, gracias al entorno de seguridad que este provee, permitiendo con ello una publicación de datos fidedigna.

El presente trabajo de diploma cumple con el objetivo general y con los objetivos específicos trazados al inicio de la investigación del módulo, obteniendo un sistema capaz de realizar la sincronización y replicación de bases de datos.

Se pudo comprobar que con la separación física se garantiza la seguridad en casi todos los aspectos, pero que también presenta dificultades como la necesidad de sincronizar las bases de datos ante la tarea de publicar información, dicha dificultad fue eliminada gracias a la incorporación del presente módulo al Software para la Interconexión de Redes Aisladas CID-555.

El módulo incorporado realiza las siguientes operaciones:

1. El administrador configura el Reko para la sincronización de las bases de datos, especificando, el tipo de base de datos a configurar, el puerto que se va a utilizar y la dirección donde se va a guardar el script generado por este con los cambios realizados en la base de datos principal.
2. El sistema obtiene el script de la carpeta donde el Reko lo generó y lo copia en el dispositivo de almacenamiento USB.
3. El dispositivo hace la conmutación de los dispositivos de almacenamiento USB, y el sistema guarda el script en el servidor réplica.
4. El Reko obtiene el script con los cambios en la base de datos principal, y actualiza la base de datos réplica.

Con la realización de este trabajo de diploma se logró la sincronización de datos de dos bases de datos aisladas físicamente entre sí.

ANEXOS

Anexo 1: CU 1. Mostrar Sucesos del Sistema

Objetivo	El objetivo del caso de uso es poder monitorear todos los eventos generados por el sistema ante diferentes situaciones.	
Actores	Administrador	
Resumen	El caso de uso le muestra al administrador el estado de los sucesos del sistema.	
Complejidad	Media	
Prioridad	Secundario	
Precondiciones	El usuario debe estar autenticado en el sistema.	
Postcondiciones	Muestra todos los sucesos del sistema.	
Flujo de eventos		
Flujo básico <Mostrar sucesos del sistema>		
	Actor	Sistema
	El administrador selecciona la opción *"Sucesos del sistema".*	
		El sistema muestra una interfaz devisualización donde muestra los sucesos del sistema con los siguientes campos: Fecha, Tipo de evento, y Descripción, con las siguientes opciones: Botón: *"Eliminar fila seleccionada"*.

		Botón: *"Buscar datos"*.
	El administrador selecciona uno de los servicios disponibles para los*"Sucesos del sistema".* Si elige *"Eliminar fila seleccionada"*, ir a la Sección 1: *"Eliminar fila seleccionada".* Si elige *"Buscar datos"*, ir a la Sección 2: *"Buscar datos"*.	
Sección 1: "Eliminar fila seleccionada"		
Flujo básico < Eliminar fila seleccionada>		
	Actor	Sistema
	El administrador presiona el botón *"Eliminar fila seleccionada"*	
		El sistema comprueba que haya una o varias filas seleccionadas.
		El sistema presenta un cuadro de diálogo de confirmación: ¿Desea eliminar los registros seleccionados? Con los botones "Eliminar" y "Cancelar".
	El administrador presiona el botón *"Eliminar".*	
		El sistema elimina el registro de la base de datos.
	El administrador presiona el botón	

	"Cancelar".	
		El sistema envía al actor de vuelta al paso 2 del flujo básico **< Mostrar sucesos del sistema>.**
Flujos alternos		
2º Evento <Si no hay una o varias filas seleccionadas>		
	Actor	Sistema
1.		El sistema presenta un aviso: *"*Seleccione una fila*"*.
		El sistema envía al actor de vuelta al paso 2 del flujo básico **< Mostrar sucesos del sistema>.**
Sección 2: "*Buscar datos*"		
Flujo básico < *Buscar datos*>		
	Actor	Sistema
	El administrador presiona el botón *"Buscar datos"*	
		El sistema presenta un cuadro de diálogo con las siguientes opciones de búsqueda: Campo donde se desea realizar la misma, el criterio de búsqueda y el dato a buscar.
		El sistema muestra todas las coincidencias con las búsquedas

		realizadas.
Relaciones	CU Incluidos	No procede
	CU Extendidos	No procede
Requisitos no funcionales	[Son los requisitos no funcionales específicos para el caso de uso]	
Asuntos pendientes	[Posibles mejoras al caso de uso.]	

Anexo 2: CU 2. Configurar Módulo de Sincronización de BD

Objetivo	Configurar la sincronización de las BD auxiliándose del REKO	
Actores	Administrador	
Resumen	El administrador establece los parámetros de conexión con la BD a replicar y con la BD a sincronizar. Establece la dirección donde se guardarán los archivos de sincronización.	
Complejidad	Baja	
Prioridad	Alta	
Precondiciones	El usuario debe estar autenticado. Deben existir las BD a replicar y sincronizar. Debe estar instalado el REKO.	
Postcondiciones	Muestra la interfaz de REKO.	
Flujo de eventos		
Flujo básico < Configurar el Sistema >		
	Actor	Sistema
	El administrador selecciona el módulo de configuración de BD.	

		El sistema muestra una interfaz con la opción de entrar a la interfaz de administración del REKO.
	El administrador selecciona la opción para abrir la interfaz del REKO.	
		El sistema muestra la interfaz del REKO.
Flujos alternos		
4º Evento <Los datos son incorrectos>		
	Actor	Sistema

Relaciones	CU Incluidos	CU 8. Configurar el REKO para la Sincronización de BD.
	CU Extendidos	
Requisitos no funcionales		
Asuntos pendientes		

Anexo 3: CU 3. Configurar el REKO para la Sincronización de BD

Objetivo	Establecer la BD a replicar, y BD de réplica, además de las carpetas donde se copiarán los archivos de réplica.
Actores	Administrador
Resumen	El administrador establece los parámetros de conexión con la BD a

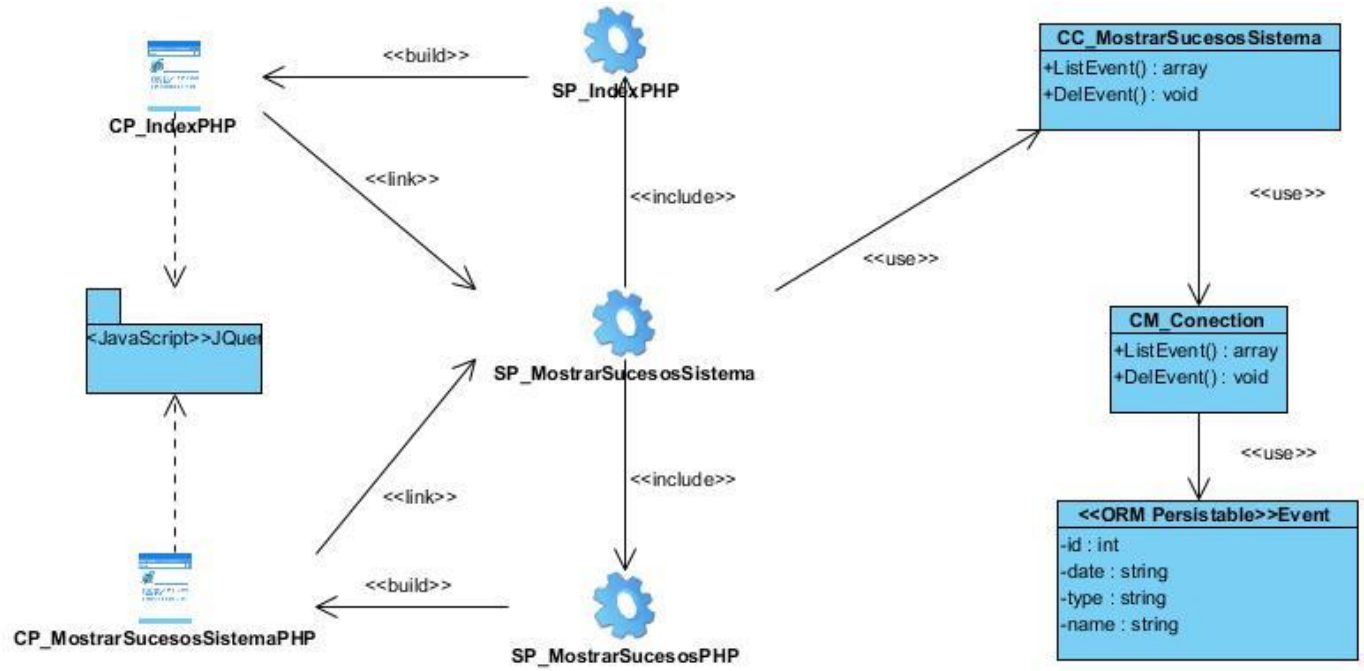
	replicar y con la BD a sincronizar. Establece la dirección donde se guardarán los archivos de sincronización.
Complejidad	Media
Prioridad	Alta
Precondiciones	El usuario debe estar autenticado. Deben existir las BD a replicar y sincronizar. Debe estar instalado el REKO.
Postcondiciones	Muestra la sincronización de las BD configurada.
Flujo de eventos	
Flujo básico < Configurar el Sistema >	

	Actor	Sistema
	El administrador se autentica en el REKO.	
		El sistema muestra la interfaz el panel de control del REKO.
	El administrador selecciona la opcion de configurar la sincronización con el par de BD a sincronizar y replicar.	
		El sistema muestra las opciones de configurar los parámetros de conexión a las BD a sincronizar.
	El administrador establece los parámetros de conexión para cada una de las BD. 1- Nombre de la BD. 2- Usuario. 3- Contraseña. 4- Dirección del servidor.	

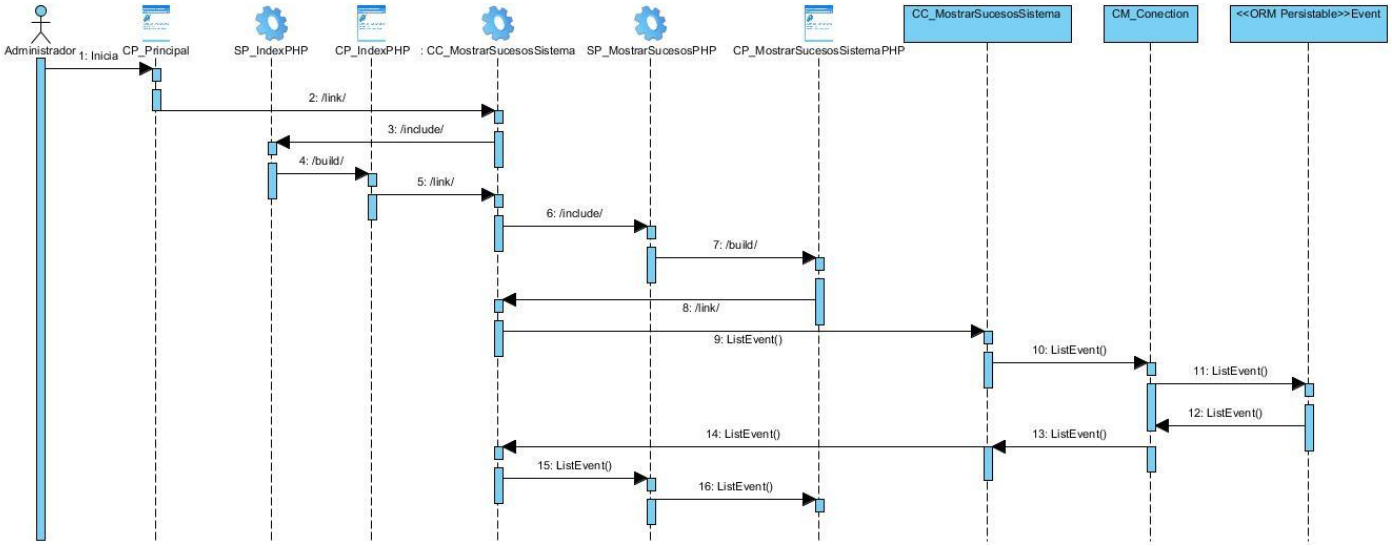
	5- Puerto de servicio de BD.	
		El sistema muestra mensaje de confirmación.
Flujos alternos		
4º Evento <Los datos son incorrectos>		
	Actor	Sistema

Relaciones	CU Incluidos	
	CU Extendidos	
Requisitos no funcionales		
Asuntos pendientes		

Anexo 4: DCD Mostrar Sucesos del Sistema. Módulo Sincronización de Bases de Datos.



Anexo 4: Diagrama de Secuencia. Mostrar Sucesos del Sistema. Módulo Sincronización de Bases de Datos.



Caso de Prueba # 1:

Condiciones de ejecución: Debe ser ejecutado como administrador en Windows o root en caso de Linux.

		Configuración del Servidor FTP						Gestor de Base de Datos a Utilizar		Configuración del Servidor de Bases de Datos						
Escenario	**Descripción**	**Ip**	**Puerto**	**Usuario**	**Contraseña**	**Directorio de Descarga**	**Directorio Binario Local**	**MY SQL**	**Postgresql**	**Ip**	**Puerto**	**Nombre**	**Usuario de Conexión a la BD**	**Contraseña**	**Respuesta del sistema**	**Flujo central**
EC 1.1 Inicio del replicador	Se ejecuta el replicador sin ser administrador del sistema	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	El replicador muestra un mensaje de error (Debe ejecutar el software como administrador)	Ejecutar el replicador sin ser administrador, superusuario o root en el caso de Linux.
	Se ejecuta el replicador siendo administrador del sistema	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	El replicador muestra su interfaz para comenzar la replicador	Se ejecuta el replicador siendo administrador, superusuario o root en el caso de Linux y comienza la configuración de los parámetro de inicio de Reko.

Tabla 2. Caso de Uso: Configuración del Replicador Reko.

REFERENCIAS BIBLIOGRÁFICAS

[1] "Kioskea - Comunidad informática." [Online]. Available: http://es.kioskea.net/. [Accessed: 16-May-2013].

[2] "Microsoft Access 2000. Qué es una base de datos." [Online]. Available: http://www.emagister.com/curso-microsoft-access-2000/que-es-base-datos. [Accessed: 09-Jun-2013].

[3] "Replicación de SQL Server." [Online]. Available: http://msdn.microsoft.com/es-es/library/ms151198.aspx. [Accessed: 16-May-2013].

[4] Zapata Pérez. Yasmani, "Estrategia para la replicación de datos espaciales en Sistemas de Información Geográfica.," UCI, La Habana, 2008.

[5] Banguela, Roberto Carlos Diéguez Sánchez y Sergio Fernández, "Sistema de sincronización y gestión de nodos aislados para la replicación de bases de datos de PostgreSQL.," UCI, La Habana, 2008.

[6] Hernández Vega. Carlos Manuel, "CID-555 Mail Gateway for intercommunication between isolated networks.," 2012.

[7] Meinel, F. y Cheng, C., "Research on the Lock-Keeper technology: Architectures, applications and advancements.," 2004.

[8] "CronSQL Herramienta de sincronización de bases de datos." [Online]. Available: http://www.spuch.com/productos/cronsql/cronsql.htm. [Accessed: 16-May-2013].

[9] "About SQLITE." [Online]. Available: http://sqlite.org/about.html. [Accessed: 09-Jun-2013].

[10] Bedoya López. Bárbara Daysi y Morales Vazquez. Yosbel, "Software para la interconexión de redes aisladas.Módulo correo electrónico V2.0," UCI, La Habana, 2012.

[11] "Manual de Ice." Copyright © 2011, ZeroC, Inc.

[12] "ZeroC - The Internet Communications Engine (Ice)." [Online]. Available: http://www.zeroc.com/ice.html. [Accessed: 16-May-2013].

[13] González Duque. Raúl, "Python para todos".

[14] "Introducción, definición y evolución de PHP. ¿Qué es PHP? Definición de PHP." [Online]. Available: http://php.ciberaula.com/articulo/introduccion_php. [Accessed: 16-May-2013].

[15] "Diccionario informática - HTML, Definición." [Online]. Available: http://internet-ka.com/Diccionario/H-Html.htm. [Accessed: 16-May-2013].

[16] White. Stephen A., "Introduction to BPMN." 2006.

[17] Ortiz Noriega. D y López Boullon. C. J, “Modulo Reko para la resolución de conflictos.” 2010.

[18] Devillart, N. ESO C, “Library for an Image Processing Software Environment.” 2001.

[19] B. Bibeault and Y. Katz, “jQuery in Action.” Manning Publications, 2010.

[20] “Rijndael.” [Online]. Available: http://msdn.microsft.com/es-es/library/system.security.cryptography.rijndael.aspx. [Accessed: 09-May-2013].

[21] Pressman Roger S. “Un toque practico. Quinta Edición.”

[22] “MeRinde - Implementación.” [Online]. Available: http://merinde.net/index.php?option=com_content&task=view&id=138&Itemid=193. [Accessed: 16-May-2013].

[23] “Exposición de diagrama de despliegue.” [Online]. Available: http://www.calameo.com/books/001765447a014cba1733c. [Accessed: 16-May-2013].

[24] “Diagramas de componentes de UML: Referencia.” [Online]. Available: http://msdn.microsoft.com/es-es/library/dd409390.aspx. [Accessed: 09-Jun-2013].

BIBLIOGRAFÍA CONSULTADA

1. **Kioskea - Comunidad informática.** [En línea] http://es.kioskea.net
2. **Microsoft Access 2000. Qué es una base de datos.** [En línea]. http://www.emagister.com/curso-microsoft-access-2000/que-es-base-datos.
3. **Replicación de SQL Server.** [En línea]. http://msdn.microsoft.com/es-es/library/ms151198.aspx.
4. **Zapata Pérez. Yasmani**, "Estrategia para la replicación de datos espaciales en Sistemas de Información Geográfica.," UCI, La Habana, 2008.
5. **Banguela, Roberto Carlos Diéguez Sánchez y Sergio Fernández,** "Sistema de sincronización y gestión de nodos aislados para la replicación de bases de datos de PostgreSQL.," UCI, La Habana, 2008.
6. **Hernández Vega. Carlos Manuel,** "CID-555 Mail Gateway for intercommunication between isolated networks.," 2012.
7. **Meinel. C. y Cheng. F,** "Research on the Lock-Keeper technology: Architectures, applications and advancements.," 2004.
8. **CronSQL Herramienta de sincronización de bases de datos.** [En línea]. http://www.spuch.com/productos/cronsql/cronsql.htm.
9. **About SQLITE.** [En línea]. http://sqlite.org/about.html.
10. **Bedoya López. Bárbara Daysi y Morales Vazquez. Yosbel,** "Software para la interconexión de redes aisladas.Módulo correo electrónico V2.0," UCI, La Habana, 2012.
11. **Manual de Ice.** Copyright © 2011, ZeroC, Inc.
12. **ZeroC - The Internet Communications Engine (Ice).** [En línea]. http://www.zeroc.com/ice.html.
13. **González Duque. Raúl,** "Python para todos".
14. **Introducción, definición y evolución de PHP. ¿Qué es PHP? Definición de PHP.** [En línea].http://php.ciberaula.com/articulo/introduccion_php.
15. **Diccionario informática - HTML, Definición.** [En línea]. http://internet-ka.com/Diccionario/H-Html.htm.
16. **White. Stephen A.,** "Introduction to BPMN." 2006.
17. **Ortiz Noriega. D y López Boullon. C. J.** "Modulo Reko para la resolución de conflictos." 2010.
18. **Devillart, N. ESO C,** "Library for an Image Processing Software Environment." 2001.

19. **B. Bibeault and Y. Katz,** “jQuery in Action.” Manning Publications, 2010.
20. **Rijndael.** [En línea]. http://msdn.microsft.com/es-es/library/system.security.cryptography.rijndael.aspx.
21. **Pressman Roger S.** “Un toque practico. Quinta Edición.”
22. **MeRinde - Implementación.** [En línea]. http://merinde.net/index.php?option=com_content&task=view&id=138&Itemid=193.
23. **Exposición de diagrama de despliegue.** [En línea]. http://www.calameo.com/books/001765447a014cba1733c.
24. **Diagramas de componentes de UML: Referencia.** [En línea]. http://msdn.microsoft.com/es-es/library/dd409390.aspx.
25. **Alfonso Valdés. Javier, Gómez Peña. Ángela Gloria y Pimentel González. Luis Alberto.** “Desarrollo del módulo para la transmisión de datos de gran tamaño para el sistema Reko.” Universidad de las Ciencias Informáticas. Ciudad de la Habana: s.n., 2011.
26. **Ramírez Concepción. R,** «Sistema de sincronización y gestión de nodos aislados para la Replicación de bases de datos en PostgresSQL», 2008.
27. **K. E. Iverson**, «A programming language», in Proceedings of the May 1-3, 1962, spring joint computer conference, 1962, pp. 345–351.
28. **Horkheimer. M,** Eclipse of reason, vol. 1. Continuum Intl Pub Group, 1974.
29. **Cheng. F, Roschke. S, y Meinel. C,** «Implementing IDS Management on Lock-Keeper», Information Security Practice and Experience, pp. 360–371, 2009.
30. **Alvarez. Rubén**, «Introducción a la programación en PHP», 01-ene-2001. [En línea]. http://www.desarrolloweb.com/articulos/303.php.
31. **PHP: Hypertext Preprocessor**, 2012. [En línea]. http://www.php.net.
32. **C. Ghezzi y M. Jazayeri**, Programming language concepts. John Wiley & Sons, Inc., 1997.
33. **M. F. Sanner**, «Python: a programming language for software integration and development», J Mol Graph Model, vol. 17, no. 1, pp. 57–61, 1999.
34. **I. Murwantara**, «Rational Unified Process», 2004.
35. **Marticorena. Raúl, López. Carlos y Crespo. Yania,** «Estudio de la distribución docente de pruebas del software y re-factoring para la incorporación de metodologías ágiles», [En línea].

http://scholar.googleusercontent.com/scholar?q=cache:UBhJbvHscQcJ:scholar.google.com/+definicion+%2B+casos+de+prueba+%2B+ISW&hl=es&as_sdt=0,5.

36. **SharePlex: Alto Rendimiento y Alta Disponibilidad en la Replicación de Oracle.** [En línea]. http://www.quest.com/webcast-ondemand/shareplex-alto-rendimineto-y-alta-disponibilidad-en-la-replicacin-de-o817031.aspx