Universidad de las Ciencias Informáticas

Facultad 1

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN INFORMÁTICA

Sistema de sincronización y gestión de nodos aislados para la Replicación de bases de datos en PostgresSQL.

AUTORES

Roberto Carlos Dieguez Sánchez.
Sergio Fernández Banguela

TUTOR

Mayor Rolando Ramírez Concepción

Ciudad de la Habana, julio de 2008.
"Año 50 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Roberto Carlos Dieguez Sánchez

_____

Sergio Fernández Banguela

_____

Mayor Rolando Ramírez Concepción

_____

"No hay mas que asomarse a las puertas de la tecnología y la ciencia contemporánea para preguntarnos si es posible vivir y conocer ese mundo del futuro sin un enorme caudal de preparación y conocimientos"

Fidel Castro Ruz.

AGRADECIMIENTOS

A mi madre Mayra Sánchez y mi padre Carlos Diéguez por confiar siempre en mis logros. Por haberme enseñado como alcanzar cada meta que se presenta.
A mis abuelos Inés María Rodríguez y Roberto Victo Sánchez por haberme cuidado siempre y por todo el apoyo que han dado en cada momento de mi vida.
A mi novia Rosa por su atención infinita y por todo el amor que me brinda.
A Anaelis Figueredo por todo su cariño.
A todas las personas que en lo personal me han ayudado en la realización de este trabajo a lo largo de estos meses.

Roberto Carlos Dieguez Sánchez.

A mi madre por ser única e inmensamente grande en los buenos y malos momentos.
A mi padre por la seguridad y la firmeza transmitida.
A mi abuelo porque es el libro por donde aprendo de la vida.
Y a mi mujer por su apoyo, su paciencia y su confianza.

Sergio Fernández Banguela.

A aquellas personas que de una forma u otra han aportado ideas para el desarrollo de este trabajo de diploma.
A los compañeros de estudio con los que hemos compartido valiosas experiencias a lo largo de la carrera.
A nuestros familiares por todo el apoyo y la confianza depositada.
A Rolando Ramírez Concepción, Norge Fajardo Vega y Dayana Méndez Alayo.
A la revolución cubana, entre tantas cosas, por darnos la posibilidad de ingresar a la educación superior y optar por un título universitario.

Sergio y Roberto Carlos.

DEDICATORIA

En especial quiero dedicar este trabajo a mi abuelo Roberto Víctor Sánchez.
A mis padres, a Anaelis, mis abuelos Eduardo Dieguez y Luisa Valdez que aun lejos me han apoyado, a mis hermanos Beatriz, Claudia, Patricia, Maryulis y Carlos Alberto. A mi primo Landy que es mi otro hermano. A mi novia se lo dedico con todo mi corazón. A mis tíos Eduardo, Michel, Jorge, Yarida, Aliria, Diana y Merlin; y a mi prima Marden que mi ejemplo le guie a obtener resultados gratos.

Roberto Carlos Dieguez Sánchez.

A mi madre, mi padre, mi hermana, mi abuelo Hector, mi abuela Margarita, mi abuelo Sergio, mi abuela Caridad, mi mujer… A Elena, Salvador, Yailén, Maikel, Osmary, Yilian, Hector Carbera, Ramiro, Javier, Laura, Hector Valdés, Lianis, Mailín, Marisel, Quintana, Domingo, Rosario, Leidy, Yordalis, Leonardo, Sinecio, Carmen, Yaser, Maricela, Bienvenida, Diana, Lázara y a mis amigos.

Sergio Fernández Banguela.

RESUMEN

Las organizaciones generan flujos informativos que condicionan el proceso de toma de decisiones. Estos flujos están asociados a determinados procesos que se desarrollan en las entidades: abastecimiento, producción, inventario, finanzas, investigación y desarrollo, entre otros.

Para los sistemas es imprescindible que el dato primario se obtenga e introduzca en aquellos lugares donde se generan. Estos datos deben recolectarse, analizarse, registrarse y trasmitirse para que constituya la base de futuras operaciones.

Hoy en día es imprescindible que los jefes y especialistas a todos los niveles cuenten con la información necesaria, en el momento preciso y el lugar adecuado, para lo cual los sistemas deben encargarse de hacer llegar este importante recurso, que es la información, a los centros de datos y servidores de los niveles ministeriales, provinciales y municipales. Con este objetivo, el pasado año se comenzó a trabajar en un software para la replicación de la información de bases de datos en PostgreSQL, obteniéndose con ello un sistema de réplica multimáster.

En este momento el MINFAR se encuentra en un proceso de conexión y organización de su red de datos, no lográndose aún, cubrir todas las necesidades de la institución. El sistema de réplica desarrollado no responde en su totalidad a las características de un entorno de no conectividad y creación sistematizada de nuevos centros de datos o servidores.

Para ello, en el presente trabajo se desarrollan un grupo de componentes que complementan la aplicación de réplica existente y posibilitarán el intercambio de información en los nodos y servidores sin conectividad a la red de datos, así como la distribución o sincronización de datos en los nodos introducidos. El sistema propuesto fue desarrollado utilizando tecnologías de software libre, como son: el lenguaje de programación Python y el SGBD PostgreSQL, aunque puede ser fácilmente extensible a otras.

PALABRAS CLAVE

FLUJOS INFORMATIVOS, REPLICACION, SINCRONIZACION, MINFAR, SERVIDOR, BASE DE DATOS, SGBD, SOFTWARE LIBRE, PYTHON, POSTGRESQL, MULTIMASTER.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

El Ministerio de las Fuerzas Armadas Revolucionarias (MINFAR) gestiona grandes volúmenes de datos organizados estructuralmente por niveles de jerarquía. La diversidad y confidencialidad de la información que se maneja requieren cada vez más elevados índices de calidad, disponibilidad y seguridad por parte de las aplicaciones informáticas encargadas de la informatización de sus procesos. Las principales entidades de las Fuerzas Armadas Revolucionarias (FAR) se encuentran ubicadas geográficamente a lo largo de todo el país, y en un nivel superior el Ministerio como órgano rector. De manera descentralizada se ejecutan todas las tareas referentes a cada área para la organización y persistencia de la información.

Una Base de Datos (BD) es un conjunto ordenado y no redundante de datos almacenados y disponibles para su posterior uso, manipulados por Sistemas Gestores de Bases de Datos (SGBD) que no son más que programas en los que se diseñan, administran, modifican y consultan las mismas. El término distribuida viene ligado al diseño del sistema. Una base de datos distribuida (BDD) es un conjunto de BD interconectadas lógicamente y similarmente estructurada de modo que con las operaciones de intercambio y sincronización de datos produzcan disponibilidad y transparencia al usuario final; son unidades virtuales, cuyas partes se almacenan físicamente en varias BD reales distintas, ubicadas en diferentes sitios.

En la actualidad existe una tendencia al creciente uso de las BDD dado el aumento de la conectividad y el acceso a las tecnologías. Las innumerables prestaciones que exigen los servicios económicos, el turismo y el comercio entre otros, hacen referencia a las ventajas de los sistemas de BDD, como por ejemplo: el aumento de la potencia de procesamiento, el trabajo en conjunto, el crecimiento incremental, menor sobrecarga de tareas, mayor flexibilidad, etc. [1]

En este tipo de diseño la fragmentación necesariamente invoca a operaciones como la replicación o propagación de la información condicionalmente o en demanda, de acuerdo a la lógica del negocio. Los sistemas de réplicas se encargan de configurar, automatizar y complementar estas tareas. Las aplicaciones informáticas del MINFAR no están exentas al tema. Por ejemplo, actualmente se utiliza el “Sistema de Réplica para bases de datos en PostgreSQL”, implementado por especialistas del MINFAR. PostgreSQL es el SGBD que soporta toda la estructura de BD de la organización.

Este referenciado sistema resuelve la automatización de la réplica de los datos a niveles superiores e inferiores del esquema implantado, Fig. 1. Para la configuración y uso del mismo se requieren de condiciones previas como la conectividad y tener los datos inicialmente sincronizados. Esta última actualmente se realiza de forma manual, enfrentándose a un conjunto de riesgos asociados a estos procesos como son: el mayor consumo de tiempo y la existencia de errores, entre otros

inconvenientes. Se refiere a sincronización como la coherencia total de un conjunto o porción de datos almacenados en BD diferentes.

Un requisito imprescindible de cualquier software que se desarrolle en interés de la defensa y seguridad del país es que debe y tiene que estar confeccionado a prueba de fallos y prever cualquier anomalía o situación extraordinaria. Puede darse el caso que en un momento y lugar determinado uno de los nodos o BD de cualquier nivel territorial o provincial se encuentre sin conectividad y correr el riesgo de la interrupción e inutilización de la información. Por motivos y situaciones innumerables que se pueden presentar, los nodos aislados por conectividad son un caso a considerar. Por estas razones gestionar estos procesos es un objetivo inmediato a solucionar.

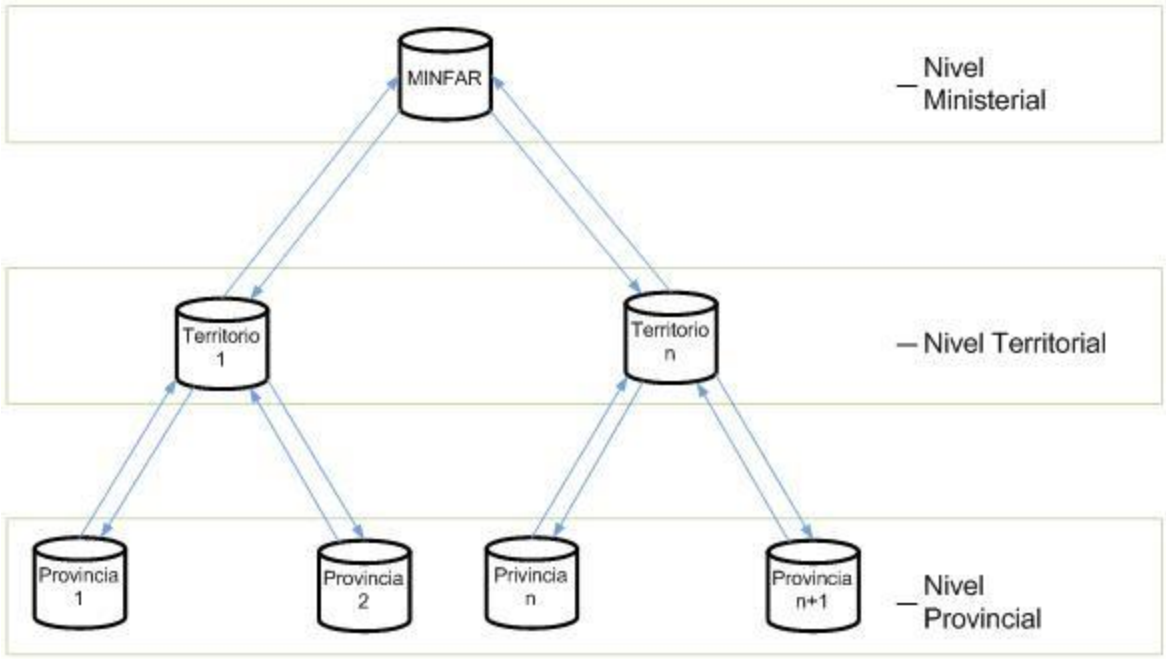


Fig. 1 Esquema jerárquico de organización de los servidores de datos.

Sintetizando la situación existente, se formula el siguiente **problema**: ¿Cómo conseguir la sincronización inicial de las bases de datos de las FAR y gestionar los nodos aislados por no conectividad en el entorno del sistema de réplica elaborado por el MINFAR para PostgreSQL? De esta forma se define el **objeto de estudio** como los procesos de réplica y sincronización de las bases de datos, y el **campo de acción** como la sincronización y gestión de nodos aislados de las bases de datos en el entorno del sistema de réplica elaborado por el MINFAR para PostgreSQL.

De acuerdo a lo descrito anteriormente, la **hipótesis** se enuncia de la siguiente manera: Si se diseñan e implementan los componentes para la sincronización inicial de las bases de datos y la gestión de nodos aislados por no conectividad en el sistema de réplica elaborado por el MINFAR para

PostgreSQL, entonces se contará con un solución integral para distribuir los datos en toda la estructura de servidores existente en las FAR.

De esta forma, se planteó el siguiente **objetivo general:** Diseñar e implementar los componentes para la sincronización inicial de las bases de datos y la gestión de nodos aislados por no conectividad en el sistema de réplica elaborado por el MINFAR para PostgreSQL.

A partir del objetivo general, con más detalles, se derivan los **objetivos específicos** que se enuncian a continuación:

1. Determinar las características básicas del sistema de réplica elaborado por el MINFAR para PostgreSQL.
2. Iniciar una labor de investigación relativa a los procesos de replicación de bases de datos haciendo énfasis en las funcionalidades de sincronización.
3. Modelar la solución para implementar la gestión de nodos sin conectividad y la sincronización de bases de datos.
4. Diseñar un sistema flexible al cambio y con calidad en sus funcionalidades.
5. Implementar los componentes diseñados para la sincronización y la gestión de nodos sin conectividad.

Para cumplir con los objetivos enunciados y demostrar la validez de la solución propuesta se enumeran las siguientes **tareas**:

1. Estudiar las características de los procesos de informatización de las FAR.
2. Estudiar la actualidad informática a nivel mundial referente a los sistemas de software y los métodos usados en el intercambio y la sincronización entre bases de datos.
3. Seleccionar las herramientas y tecnologías de diseño y construcción que son utilizadas dentro de las FAR para el desarrollo productivo.
4. Realizar el diseño del sistema utilizando la metodología y las herramientas de análisis y diseño de software escogidas.
5. Implementar el sistema utilizando las herramientas de construcción escogidas.
6. Describir un modelo de prueba eficiente.
7. Realizar las pruebas descritas en el modelo de pruebas para el sistema.

A lo largo de la investigación son utilizados métodos de investigación que ayudan a guiar exitosamente el desarrollo de la solución. Se utiliza el **método** hipotético-deductivo a partir de la hipótesis, para llegar a nuevos conocimientos y nuevas predicciones que posteriormente en el transcurso del trabajo

de diploma son sometidas a verificaciones dentro del análisis y diseño de la solución. Otro de los métodos que guían los procesos investigativos es el método histórico-lógico, que se utiliza en el análisis de la trayectoria y las etapas principales dentro de la investigación en general, expresando de forma lógica el desarrollo y los conocimientos. Dentro de la etapa de modelación de la solución se aplica el método correspondiente a la modelación constituido por abstracciones creadas para dar una explicación de la realidad existente y cuya condición principal es la relación entre el modelo y el objeto que se modela, que en este caso sería el sistema en cuestión. A lo largo del desarrollo del sistema se realizan pruebas continuas, acompañadas por entrevistas realizadas a clientes, usando los métodos empíricos generales tales como la observación y la experimentación. Todos estos métodos actúan como guía metodológica para la investigación y medición de los resultados.

En su estructura, este trabajo presenta un total de cuatro capítulos, donde el primer capítulo aborda todo lo referente a la fundamentación teórica y conceptos enunciados para el análisis y la investigación del trabajo así como una caracterización de las principales herramientas existentes.

En el segundo capítulo se presenta la solución propuesta donde se modelan los principales conceptos y entidades del sistema. Se modelan el sistema a través de los actores del sistema y los distintos casos de uso. Se hace una descripción detallada de estos casos de uso teniendo en cuenta las distintas alternativas y caminos existentes que son tomados por los actores del sistema.

En el tercer capítulo se hace un análisis del sistema y se propone el diseño para realizar la implementación. Se muestran los principales diagramas de interacción dentro del sistema y se describen las pautas para el diseño y la forma de tratar los errores que puedan surgir a lo largo del uso de la aplicación.

Por último, en el cuarto capítulo se presenta el diagrama de componentes del sistema, así como la representación de su modelo de despliegue y una descripción detallada de las pruebas realizadas a la solución final para garantizar una calidad óptima del producto.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.

1.1 Introducción.

Este capítulo enmarca los principales conceptos y tendencias sobre las cuales se sustenta la solución resultante. Se incorpora una explicación detallada de las principales herramientas consultadas y utilizadas en la propuesta tanto en el ámbito internacional como en las FAR, de modo que se expone una panorámica general de la manipulación e intercambio de información entre las BD de las aplicaciones de la institución.

1.2 Fragmentación, replicación y sincronización de datos.

Estos tres términos son parte del domino de las BDD, que fragmentan la información a disposición de las reglas del negocio y en busca de un rendimiento más óptimo, distribuyéndola convenientemente. Una vez fragmentada será condición indispensable tenerla sincronizada para proceder a replicar los datos en ambos sentidos según la configuración necesaria.

Las BD están constituidas principalmente por tablas, relaciones y funciones o procedimientos almacenados. Las tablas almacenan los datos en cada una de sus tuplas o filas, mientras que sus columnas representan los atributos, viéndolas como listados de objetos, Fig. 2. Cada atributo se define con un tipo de dato determinado. En el ejemplo, un centro contendrá un listado de personas.

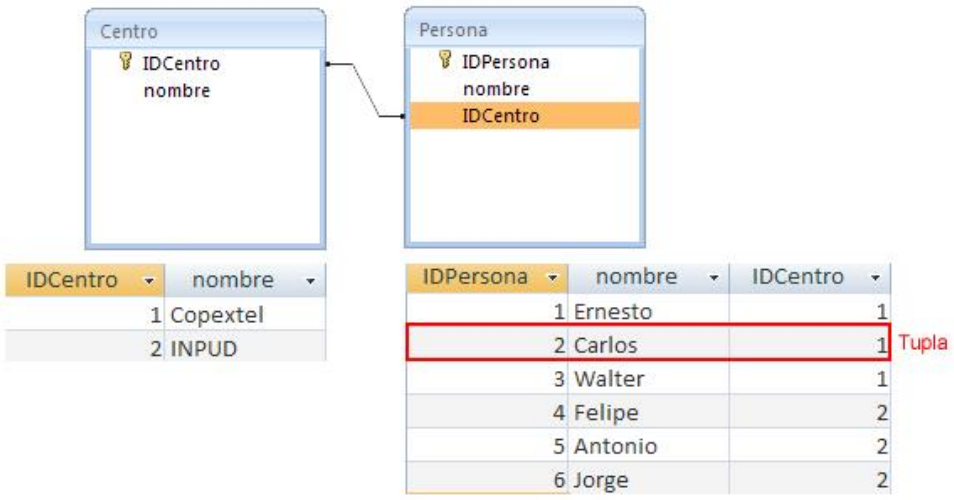


Fig. 2 Tablas y relaciones de una base de datos.

1.2.1 Fragmentación horizontal y vertical.

Para fragmentar horizontalmente los datos, una Tabla T se divide mediante operaciones de selección en subconjuntos T1, T2,…Tn. Cada fragmento se ubica en un nodo o BD, y se reconstruyen con operaciones de unión, Fig. 3.

IDPersona	nombre	primerApellido	fechaNacimiento	IDCentro
1	Ernesto	Fernández	12/1/1976	1
2	Carlos	Martín	2/15/1980	1
3	Walter	Rojas	7/14/1987	1
4	Felipe	García	3/23/1990	2
5	Antonio	Estrada	4/11/1985	2
6	Jorge	Pérez	9/20/1979	2

Tabla T ubicada en el nodo A que registra los datos del país.

IDPersona	nombre	primerApellido	fechaNacimiento	IDCentro
1	Ernesto	Fernández	12/1/1976	1
2	Carlos	Martín	2/15/1980	1
3	Walter	Rojas	7/14/1987	1

Tabla fragmento T1 ubicada en el nodo B que registra los datos del centro Copextel

IDPersona	nombre	primerApellido	fechaNacimiento	IDCentro
4	Felipe	García	3/23/1990	2
5	Antonio	Estrada	4/11/1985	2
6	Jorge	Pérez	9/20/1979	2

Tabla fragmento T2 ubicada en el nodo C que registra los datos del centro INPUD

Fig. 3 Fragmentación horizontal.

En la fragmentación vertical, el proceso de división se realiza mediante operaciones de proyección, donde cada fragmento deberá incluir la llave o identificador de la tabla T (id), Fig. 4. Este tipo de fragmentación se realiza generalmente atendiendo a la frecuencia de uso de la información, y su reconstrucción se realiza con una operación de join (palabra reservada del lenguaje de consulta, utilizada para formar criterios de selección entre dos o más tablas).

IDPersona	nombre	primerApellido	fechaNacimiento
1	Ernesto	Fernández	12/1/1976
2	Carlos	Martín	2/15/1980
3	Walter	Rojas	7/14/1987
4	Felipe	García	3/23/1990
5	Antonio	Estrada	4/11/1985
6	Jorge	Pérez	9/20/1979

Tabla fragmento T3 ubicada en el nodo D

IDPersona	nombre	IDCentro
1	Ernesto	1
2	Carlos	1
3	Walter	1
4	Felipe	2
5	Antonio	2
6	Jorge	2

Tabla fragmento T4 ubicada en el nodo E

Fig. 4 Fragmentación vertical.

Hasta ahora se han enunciado los términos que conforman una BD, pero para mayor comprensión se resume lo que es Servidor de BD: Conjunto de BD agrupadas en un mismo SGBD ubicado en una computadora. [2]

1.2.2 Entornos, técnicas y conflictos de la replicación.

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.

Básicamente existen dos tipos o entornos de réplicas: El de solo lectura o maestro-esclavo (master-slave), que permite al nodo maestro realizar consultas de lectura/escritura, mientras que los nodos esclavos solo de lectura; y el otro entorno es el par-a-par o multimáster, donde muchos nodos maestros interactúan entre si, facilitando la sincronización de los cambios.

Ambos entornos pueden aplicar un modelo de distribución asincrónica, que captura los cambios locales almacenándolos en una cola y propagándolos a los sitios remotos a intervalos regulares de tiempos; o aplicar un modelo sincrónico, que ejecuta cualquier cambio de los sitios que intervienen en el ambiente de réplica como parte de una única transacción, si falla la operación en al menos un sitio entonces se anula la transacción completa. Cada modelo usa tecnologías particulares que a su vez presentan limitaciones, Fig. 5.

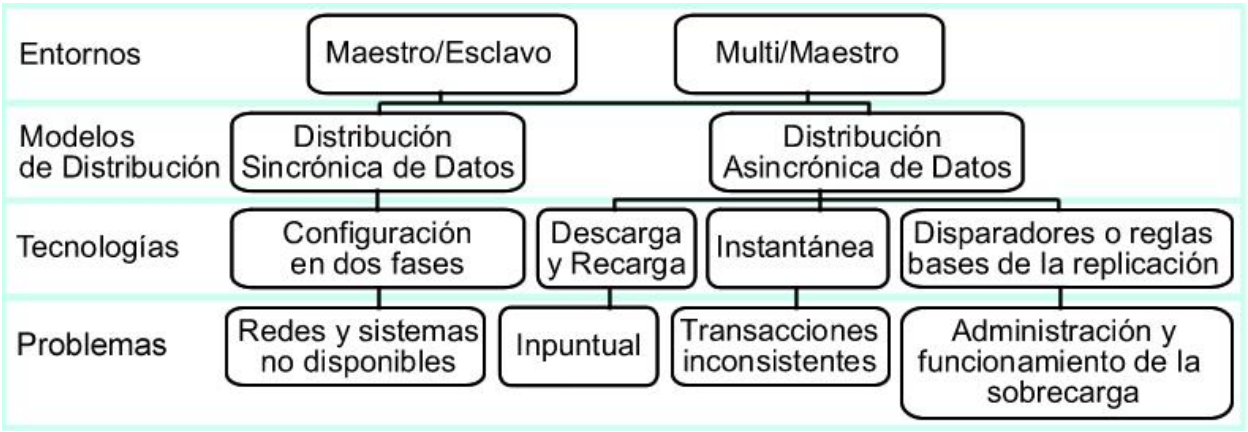


Fig. 5 Esquema de réplica.

Detallando las técnicas de replicación de la Fig. 5, se tiene que la configuración en dos fases (two phase commit) solo se aplica si todos los sistemas implicados están conectados y listos; la de descarga y recarga (dump and reload) copia la salva en un dispositivo de almacenamiento para luego distribuirla por los servidores pero en la mayoría de los casos se consultan datos que pueden tener hasta semanas de atraso, y además el proceso se realiza manualmente; la instantánea (snapshot)

hace una copia exacta de una tabla que es la que se réplica, pero la tablas esclavas son de solo lectura por lo que se recomienda aplicar cuando los datos no cambian mucho, o si los sitios están frecuentemente desconectados y es aceptable un periodo largo del tiempo de actualización; la de los disparadores (triggers) ejecuta una acción cuando ocurre un evento de inserción, modificación o eliminación.

Al aplicar la replicación pueden ocurrir conflictos que deben ser considerados y tratados, más aún si se permiten actualizaciones concurrentes en los mismos datos. Por ejemplo: El conflicto de actualización que ocurre cuando dos transacciones de distintos sitios actualizan el mismo registro en forma cercana de tiempo, y se resuelve priorizando los servidores, o considerando correcta la más nueva o vieja de acuerdo al tiempo, o garantizando que cada registro sea manipulado por un único servidor; el conflicto de unicidad que ocurre por ejemplo cuando dos transacciones de distintos sitios intentan insertar un registro con el mismo valor de la llave primaria (identificador de cada tupla), y se resuelve brindando un rango distinto de números de llaves primarias para cada servidor, o agregando a la llave el identificador del servidor, o replicando en tablas separadas; el conflicto de supresión que es cuando una transacción intenta borrar un registro y otra intenta modificarlo o borrarlo, y una posible solución es que desde los sitios replicados no se puedan borrar físicamente los datos, estos se marcarían y eliminarían posteriormente mediante una función ejecutada periódicamente en el servidor; y por último el conflicto de orden que puede ocurrir en ambientes con tres o más sitios maestros cuando uno X está bloqueado y las transacciones de modificación siguen propagándose a los otros, al final las modificaciones debieron propagarse al sitio X en orden distinto a como ocurrió en los demás, provocando conflictos, los cuales suelen resolverse aplicando distintas prioridades a los sitios maestros para ordenar las transacciones.

1.2.3 Necesidad de la sincronización inicial.

La sincronización significa coherencia, en este caso, entre los datos de las tablas que intervienen en la réplica entre dos BD. Hay un caso fundamental en el que es necesario tener inicialmente sincronizadas estas tablas y es cuando se agrega un nuevo servidor, para poder hacer operaciones de consulta, modificación y eliminación en los datos que supuestamente deben existir.

1.3 Tecnologías actuales.

1.3.1 Tendencia al software libre.

Se denomina software libre al que otorga a los usuarios las libertades de usarlo con cualquier propósito, estudiar su funcionamiento y adaptarlo a las necesidades, distribuir copias, y publicar sus

mejoras en beneficio de la comunidad. La Fundación del Software Libre (FSF), creada en 1985, se dedica a eliminar las restricciones sobre la copia, redistribución, entendimiento y modificación de los programas de computadoras promocionando el uso y desarrollo del software libre en todas las ramas de esta esfera.

“Software Libre" se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. De modo más preciso, se refiere a cuatro libertades de los usuarios del software:

1. La libertad de usar el programa, con cualquier propósito (libertad 0).
2. La libertad de estudiar cómo funciona el programa, y adaptarlo a tus necesidades (libertad 1). El acceso al código fuente es una condición previa para esto.
3. La libertad de distribuir copias (libertad 2).
4. La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. (libertad 3). El acceso al código fuente es un requisito previo para esto.

Un programa es software libre si los usuarios tienen todas estas libertades. Así pues, deberías tener la libertad de distribuir copias, sea con o sin modificaciones, sea gratis o cobrando una cantidad por la distribución, a cualquiera y a cualquier lugar. El ser libre de hacer esto significa, entre otras cosas, que no tienes que pedir o pagar permisos. [3]

1.3.1.1 Licencias para el software libre.

Una licencia es aquella autorización formal que un autor de un software da a un interesado para ejercer actos de explotación legales. Pueden existir tantas licencias como acuerdos concretos entre el autor y el licenciatario. Desde el punto de vista del software libre, existen distintas variantes del concepto o grupos de licencias:

Licencia Pública General (GNU GPL): Es una licencia creada por la FSF para proteger la distribución, modificación y uso del software. Su esencia es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan estas libertades a los usuarios. Garantiza que todas las versiones del software permanezcan bajo los términos más restrictivos de la propia licencia.

LGPL: Es una especie de versión de la anterior. La Licencia Pública General de Librerías de GNU funciona exactamente igual que GPL, excepto que permite que el Software con esta licencia sea incorporado en Software Comercial. Por ejemplo sirve para que las librerías de C de Linux sirvan para

hacer programas comerciales, ya que de otra forma sólo servirían para desarrollar Software Libre. Un programa con licencia LGPL puede pasarse a la licencia GPL, pero no al contrario.

Licencia BSD: Otorgada principalmente para los sistemas BSD (Berkeley Software Distribution, Distribución de Software Berkeley, que identifica un sistema operativo derivado del Unix y nacido a partir de las aportaciones realizadas a ese sistema por la Universidad de California en Berkeley). Esta licencia pertenece al grupo de licencias de software libre y tiene menos restricciones en comparación con otras como la GPL, estando muy cercana al dominio público. La licencia BSD al contrario de la GPL permite el uso del código fuente en software no libre. Existen opiniones que destacan que este tipo de licencia no contribuye al desarrollo de más software libre.

Licencia Apache: Es un derivado de la licencia BSD (Berkeley Systern Distribution.) Fundamentalmente ambas permiten hacer casi cualquier cosa con el código fuente, también modificarlo sin publicar esas modificaciones. Es un tipo de licencia menos restrictivo pero que funciona muy bien en casos como el de Apache.

Licencia Pública de Mozilla (MPL): Se utiliza en gran cantidad de software libre de todo tipo. Similar a la BSD pero perfeccionada, cumple con la definición de software de código abierto y con las cuatro libertades del software libre. Fue desarrollada por Netscape Communications Corporation, derivada de American Online y en la actualidad controlada por la Fundación Mozilla. [4]

Copy left: Puede considerarse opuesto al copyright (Derecho de autor: Conjunto de normas y principios que regulan los derechos morales y patrimoniales que la ley concede a los autores). Se define como un grupo de derechos de propiedad intelectual que elimina las restricciones de distribución o modificación del copyright, condicionado a que el trabajo derivado se mantenga con el mismo régimen de propiedad intelectual que el original. [5]

1.3.1.2 Open Source.

Traducido literalmente "código abierto". O sea que se puede mirar el código fuente. Por esto puede ser interpretado como un término más débil y flexible que el del software libre, y tal no es el caso. Cualquier software que tenga su código fuente disponible no es Open Source, debe permitir ser modificado y redistribuido para tener esta definición. Hasta este punto el término es perfectamente equivalente al software libre, pero desde el punto de vista filosófico es totalmente incompatible puesto que este se centra en la calidad del producto como primer punto, mientras la FSF prioriza las libertades de usuario. La Open Source Initiative (OSI) es la organización que promueve este software.

La idea que late detrás del open source es bien sencilla: Cuando los programadores en internet pueden leer, modificar y redistribuir el código fuente de un programa, éste evoluciona, se desarrolla y

mejora. Los usuarios lo adaptan a sus necesidades, corrigen sus errores a una velocidad impresionante, mayor a la aplicada en el desarrollo de software convencional o cerrado, dando como resultado la producción de un mejor software. [6]

1.3.1.3 Software gratuito (freeware).

Este se distribuye sin costo y por tiempo ilimitado. Puede o no incluir una licencia de uso que impida su modificación y venta. Además suelen liberarse versiones freeware de productos para motivar la compra de versiones posteriores más completas. No hay que asociar el término gratuito al software libre, ya que este puede estar gratuitamente disponible o al costo de la distribución a través de otros medios. El término fue acuñado en 1982 por Andrew Fluegelman, que quería distribuir un programa que había escrito, llamado *PC-Talk*, pero con el que no deseaba usar métodos tradicionales de distribución de software. Fluegelman registró el término freeware, pero esos derechos de autor ya han sido abandonados. De hecho, el método que usó para distribuir *PC-Talk* hoy se denominaría shareware. [7]

1.3.1.4 Software de dominio público.

Este tipo de software no requiere de una licencia, pues sus derechos de explotación son para toda la humanidad. Cualquiera puede hacer uso de él, siempre con fines legales y consignando su autoría original. Es aquel cuyo autor dona a todos o cuyo derecho de autor ha expirado. Si un autor condiciona su uso bajo una licencia, por muy débil que sea, ya no es dominio público. [4]

1.3.2 PostgreSQL.

PostgreSQL es un SGBD relacional orientado a objetos de software libre, liberado bajo la licencia BSD. Su desarrollo no es manejado por una sola compañía, como otros proyectos Open Source, es dirigido y desarrollado por la comunidad PGDG (PostgreSQL Global Development Group).

Está caracterizado por su alta concurrencia, pues mediante su sistema denominado Acceso Concurrente Multi Version (MVCC) permite el acceso a las tablas que están bajo un proceso que se está ejecutando en ellas sin necesidad de bloqueos. Por otra parte posee una amplia variedad de tipos nativos, entre otros son los casos de:

1. Texto de largo ilimitado.
2. Números de precisión arbitraria.
3. Figuras geométricas
4. Direcciones IP

5. Arrays
6. Otros tipos de datos creados por los usuarios (Ejemplo: GIS (Sistema de Información Geográfica)).

Integra otras características que son generales en cada uno de los distintos SGBD como:

1. Llaves ajenas o Llaves Foráneas (foreign keys).
2. Disparadores (triggers).
3. Vistas.
4. Integridad transaccional.
5. Herencia de tablas.
6. Tipos de datos y operaciones geométricas.

Los bloques de código que se ejecutan en el servidor (funciones), pueden ser escritos en varios lenguajes, desde las operaciones básicas de programación, tales como bifurcaciones y bucles, hasta las complejidades de la programación orientada a objetos o la programación funcional.
Algunos de los lenguajes que se pueden usar son los siguientes:

1. PL/PgSQL (similar al PL/SQL de Oracle).
2. C.
3. C++.
4. Gambas.
5. Java PL/Java web.
6. PL/Perl.
7. plPHP.
8. PL/Python.
9. PL/Ruby.
10. PL/Scheme. [8]

1.3.3 Sistemas de réplica existentes.

Un gran número de empresas se destacan hoy en la producción de software de herramientas para el manejo y la gestión de sistemas de base de datos (BD). En la actualidad esta rama de la informática presenta un alto nivel de importancia en el desarrollo global de las Tecnologías de la Informática y las Comunicaciones (TIC). La necesidad de datos cada día es mucho mayor, debido al marcado avance de la ciencia y la tecnología, de los estudios y los resultados. Se trata de un proceso en plena

expansión con un crecimiento de entre el 100 y el 150 por ciento cada año. Es precisamente aquí donde se cree que se va a producir la mayor parte del crecimiento de las TIC. Hay que tener en cuenta que los sistemas de servidores que dominan las arquitecturas de los ordenadores hoy en día no fueron diseñados para soportar las cantidades de datos que se generan en la actualidad. Considerándose que, según las informaciones conocidas y las previsiones analizadas, el almacenamiento empresarial está llamado a doblarse cada año, las compañías deben encontrar con rapidez una solución efectiva para el control de su almacenamiento [1]. Esto implica un crecimiento en el volumen de hardware como, por ejemplo, en los servidores de BD, así como en los sistemas para el transporte de información, y para el control y manejo de estos volúmenes de almacenamiento.

Entre estos sistemas se estudiaron los que se destacan en la replicación de datos, que son los encargados de transmitir la información distribuida entre las BD, y dentro de este tema los encargados de realizar la sincronización y el transporte de los datos en PostgreSQL.

Postgres-R, PgReplicator, Usogres, ERServer y Slony-I son sistemas de réplica para PostgresSQL, que no implementan una solución de réplica para las bases de datos fragmentadas.

1.3.3.1 Slony.

Slony-I es un sistema de replicación “maestro a múltiples esclavos” para PostgreSQL apoyado en réplica de cascadas. Por ejemplo, un nodo A puede alimentar otro nodo B que alimenta a su vez a otro C, por lo que permite a un esclavo ser a su vez maestro para otro servidor. La gran imagen para el desarrollo de este sistema es que incluye todos los rasgos y capacidades necesarios para reproducir grandes BD a un número razonable y limitado de sistemas de esclavos (aproximadamente 12 esclavos inmediatos). Está diseñado para ser usado en centros de datos y sitios auxiliares dónde el modo normal de funcionamiento es que todos los nodos estén disponibles todo el tiempo y donde todos puedan asegurarse. Es por este motivo que, aunque está demostrada su tolerancia a fallos, no se aconseja usarse cuando se tienen nodos que probablemente se puedan quedar fuera de la red, que estén incluidos dentro de una red inestable o en sitios donde la configuración cambia de manera casual. Slony-I realiza una réplica de espejos, exactamente igual al origen de datos.

Actualmente se está desarrollando Slony-II que a diferencia de este será multimáster y presentará muchas características a considerar. [9]

1.3.3.2 PgCluster.

Es un sistema de replicación que incluye balanceo de carga. Es sincrónico, aplica un modelo de distribución multimáster para PostgreSQL, y tiene la capacidad de sincronizar el árbol de directorios.

Funciona dentro de la misma base de datos, replicando los datos por consultas (funciones de BD que devuelven reportes condicionados). Incluye tolerancia a fallos y está publicado bajo licencia BSD. [10]

1.3.3.3 Sistema de réplica para bases de datos distribuidas en PostgreSQL.

Este sistema, desarrollado por el MINFAR en el 2007, presenta una solución de réplica para bases de datos fragmentadas. Elaborado para el SGBD PostgresSQL, trabaja en conjunto con Slony-I, y es el que actualmente resuelve los casos de replicación en los centros que pertenecen a las FAR en toda Cuba. Su principio de funcionamiento está basado en trigger y apoyado en Slony que realiza el transporte en forma de espejo de las tablas maestras a las esclavas; o sea, si se inserta en una maestra, Slony transporta fielmente el cambio e inserta en la esclava y de igual manera si se elimina. En la Fig. 6 se muestra el principio de funcionamiento del Sistema de Réplica para PostgreSQL existente en las FAR.

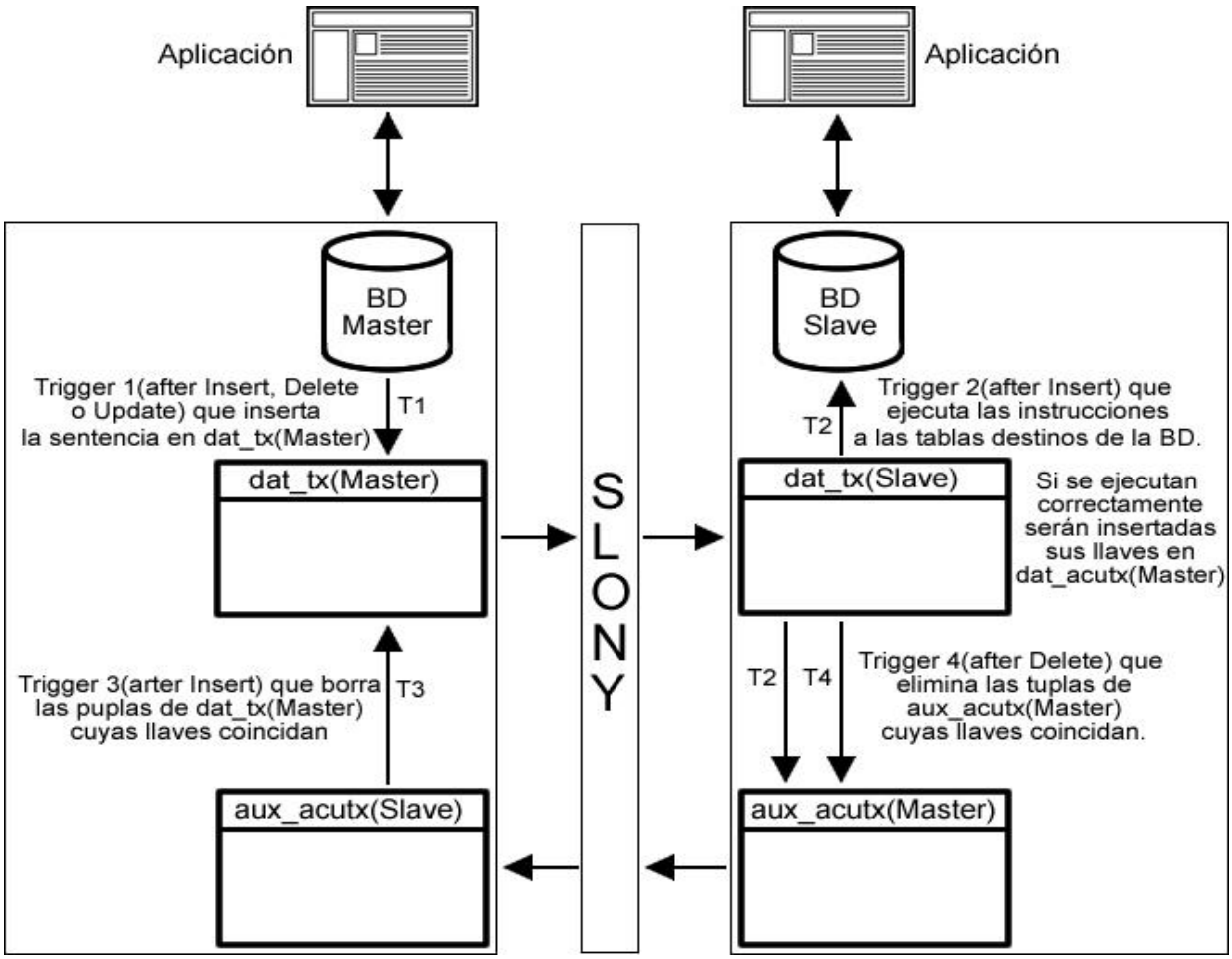


Fig. 6 Principio de funcionamiento del Sistema de réplica para bases de datos distribuidas en PostgreSQL.

1.3.4 Python.

Python es un lenguaje multiplataforma y orientado a objetos con soporte para cualquier tipo de programa, desde aplicaciones de escritorio a servidores de red o incluso páginas web a pesar de no ser creado específicamente para esta funcionalidad. Es interpretado, lo que significa que no se necesita compilar el código fuente para poder ejecutarlo (esta se realiza de manera transparente para el programador), lo que ofrece ventajas como la rapidez de desarrollo e inconvenientes como una menor velocidad. Originalmente se desarrolló para Unix, aunque cualquier sistema es compatible con el lenguaje siempre y cuando exista un intérprete programado para él.

Tiene una sintaxis muy visual, gracias a una notación con márgenes de obligado cumplimiento. Esto ayuda a que todos los programadores adopten las mismas notaciones y que los programas de cualquier persona tengan un aspecto muy similar.

En los últimos años el lenguaje se ha hecho muy popular, gracias a varias razones como:

1. La cantidad de librerías que contiene, tipos de datos y funciones incorporadas en el propio lenguaje, que ayudan a realizar muchas tareas habituales sin necesidad de tener que programarlas desde cero.
2. Un programa puede tener de 3 a 5 líneas de código menos que su equivalente en Java o C, facilitando la sencillez y velocidad con la que se crean los programas
3. La cantidad de plataformas en las que se puede desarrollar, como Unix, Windows, OS/2, Mac, Amiga y otros Sistemas Operativos.
4. Además, es gratuito, incluso para propósitos empresariales.

Finalmente está en movimiento y en pleno desarrollo, pero ya es una interesante opción para realizar todo tipo de programas que se ejecuten en cualquier máquina. [11]

1.3.5 Interfaz de usuario. GTK. Glade.

GTK+ (GIMP toolkit) es un grupo de herramientas para General Image Manipulation Program (GIMP). Conjunto de bibliotecas para desarrollar interfaces gráficas de usuario (GUI). Es software libre bajo la licencia GPL y multiplataforma. Fue creado para desarrollar el programa de manejo de imágenes GIMP y actualmente es muy usado por muchos otros programas en los sistemas GNU/Linux, o en otras plataformas, permitiendo lenguajes de programación como C, C++, C#, Java, Perl, PHP o Python.
GTK está basado en tres bibliotecas:

1. **GLib:** Biblioteca de bajo nivel que proporciona manejo de estructura de datos para C, interfaces para funcionalidades de tiempo de ejecución como ciclos, hilos, carga dinámica o un sistema de objetos.
2. **Pango:** Biblioteca para el diseño y renderizado de texto. Es el núcleo para manejar las fuentes y el texto de GTK+2.
3. **ATK:** Biblioteca para crear interfaces de una gran accesibilidad y muy importante para personas discapacitadas. Pueden usarse útiles como lupas de aumento, lectores de pantalla, o entradas de datos alternativas. [12]

Existe un vínculo de esta biblioteca gráfica con el lenguaje Python llamado PyGTK que permite la construcción de aplicaciones gráficas potentes. [13]
Glade es una herramienta de desarrollo visual de interfaces gráficas mediante GTK bajo GPL. Es independiente del lenguaje de programación y desde su versión Glade-3 no genera código fuente sino un fichero XML (Extensible Markup Language), que usando la biblioteca de librerías libglade puede ser

cargado dinámicamente por aplicaciones que necesiten su uso. Con libglade, los archivos XML de Glade pueden ser usados en numerosos lenguajes de programación como son el caso de C, C++, Java, Perl, Python, C#, Pike, Ruby, Haskell, entre otros. [14]

1.3.6 Mozilla Firefox.

Es un navegador de Internet, con interfaz gráfica de usuario, desarrollado por la Corporación Mozilla y un gran número de voluntarios externos. Es software libre y está bajo la licencia MPL, destacándose entre los más populares de su tipo por su rapidez, seguridad y facilidad de personalización. La Corporación Mozilla es una empresa filial de propiedad total de la Fundación Mozilla, que coordina e integra el desarrollo de aplicaciones informáticas relacionadas con Internet, y se dedica a la creación de software libre.

1.3.7 Proceso Unificado de Rational (RUP).

Para el desarrollo de esta propuesta se opta por la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos a nivel mundial El Proceso Unificado de Rational (RUP).

RUP disciplina las formas de asignar tareas y responsabilidades, detallando quién hace qué, cuándo y cómo lo hace. Practica un desarrollo iterativo e incremental, administra los requisitos y controla los cambios realizados. Además usa una arquitectura basada en componentes y modela visualmente el software tratado. Esta metodología esta centrada en la arquitectura y guiada por Casos de Uso. Incluye artefactos y roles. Su uso arroja resultados provechosos en cualquier estructura de proyectos, marcando una mayor organización de las actividades y procesos dentro de los flujos de trabajo y sus respectivas fases, siendo la más apropiada para el sistema propuesto. [15]

1.3.7.1 Lenguaje Unificado de Modelado (UML).

UML se usa para las representaciones visuales de la arquitectura. Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad, aún cuando todavía no es un estándar oficial. Se usa para visualizar, especificar, construir y documentar un sistema de software. Ofrece un estándar para describir un plano del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables. Entre estos componentes están los diagramas de estructura, de comportamiento y de interacción, que enfatizan en

los elementos que deben existir en el sistema modelado y lo que debe suceder sobre el flujo de control y de datos entre los elementos de dicho sistema. [16]

1.3.7.2 Visual Paradigm.

Visual Paradigm para UML es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software como son el análisis y diseño orientados a objetos, la construcción, las pruebas y el despliegue. Este software de modelado ayuda a una construcción más rápida de aplicaciones, con calidad y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. [17]

Visual Paradigm ofrece:

1. Entorno de creación de diagramas para UML 2.0.
2. Diseño centrado en casos de uso y enfocado al negocio, que generan un software de mayor calidad.
3. Uso de un lenguaje estándar, común para todo el equipo de desarrollo, que facilita la comunicación.
4. Capacidades de ingeniería directa e inversa.
5. Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
6. Disponibilidad de múltiples versiones, para cada necesidad.
7. Disponibilidad de integrarse en los principales IDEs.
8. Disponibilidad en múltiples plataformas. [18][19]

1.4 Conclusiones.

Como resultado de la investigación realizada y adecuándose a las políticas de usabilidad y seguridad de las FAR, se decidió seguir la línea de implementación trazada por el software actualmente usado para la replicación: Sistema de réplica para bases de datos distribuidas en PostgreSQL. El lenguaje de programación que se usará para el sistema es Python. Se usarán las librerías gráficas de GTK y la herramienta de desarrollo visual de interfaces Glade. Los reportes necesarios se mostrarán en formato HTML mediante el navegador web Mozilla Firefox. El proceso será guiado por la metodología de desarrollo RUP y los diagramas serán desarrollados por Visual Paradigm para UML.

Para la base del conocimiento se abordaron todos los conceptos y terminologías involucradas en los procesos de réplicas de BD y en su automatización en el ámbito de las FAR. También fueron detallados puntos como son las tendencias actuales, las licencias de software y los sistemas similares existentes.

CAPÍTULO 2: PRESENTACIÓN DE LA SOLUCIÓN PROPUESTA.

2.1 Introducción.

En este capítulo se justifica y enuncia el Modelo de Dominio como una vía eficiente de representar el negocio de la solución. Conceptualizando las principales entidades y sus relaciones, y definiendo las necesidades y principales funcionalidades del sistema. Se modela 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. Teniendo en cuenta las alternativas que puede tomar el actor se han descrito muy bien los casos de uso del sistema.

2.2 Propuesta de solución.

Para la solución de la propuesta se presenta un sistema para sincronizar bases de datos en PostgreSQL con la funcionalidad de gestionar todo el proceso de intercambio de información entre nodos sin conectividad, aplicado al marco de las FAR. Es un complemento del software actualmente utilizado por el Ministerio para la replicación de sus servidores de datos, por lo que se hace uso de los mismos principios, modelos y técnicas de réplica.

2.3 Modelo de Dominio.

Esta aplicación tiene poca estructuración dentro del negocio, y solo se pueden identificar conceptos y objetos relacionados entre si dentro del área de interés, por lo tanto es más factible realizar un Modelo de Dominio para representar estas clases conceptuales (ideas u objetos físicos). Este modelo deja bien claro como funciona el entorno en el cual está enmarcado el problema. Para cualquier solución de casos de uso, los conceptos e ideas propias del dominio del problema serán las mismas.

En el Modelo de Dominio se explica con un lenguaje común la relación entre los objetos que forman parte del negocio de la aplicación, destacándose el sistema de replicación al cual se ha hecho referencia anteriormente.

2.3.1 Diagrama del Modelo de Dominio.

En la fig. 7 se muestran los principales conceptos identificados dentro del modelo de dominio del sistema y las relaciones existentes entre cada uno de ellos.

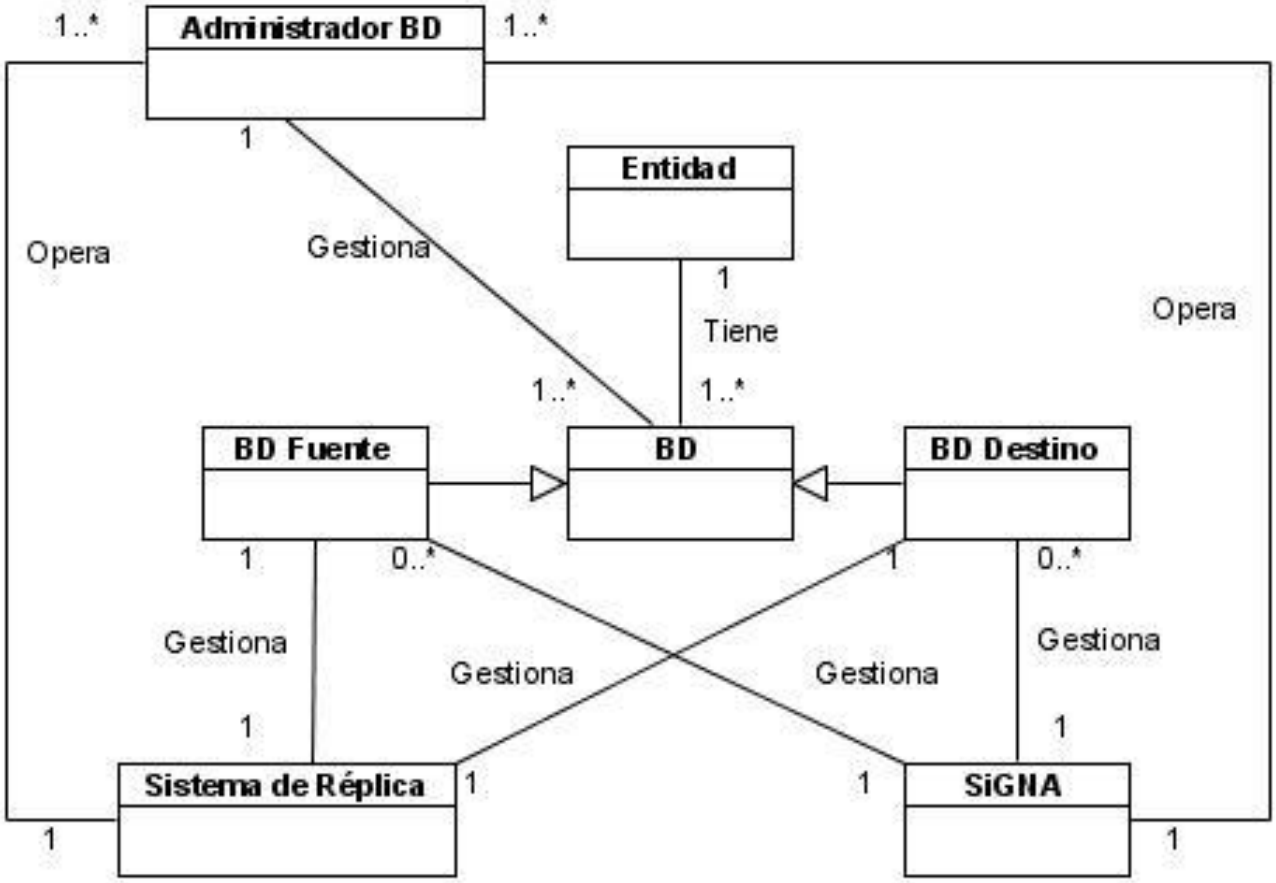


Fig. 7 Modelo de Dominio.

2.3.2 Glosario de conceptos del Modelo de Dominio.

Entidad: Representa a las dependencias de la Institución, tales como regiones, sectores militares, unidades militares, etc.

BD: Base de datos asociada al sistema.

BD Fuente: Base de datos que contiene la configuración y el origen de la información a replicar.

BD Destino: Base de datos a donde se transmite la información a ejecutar por el sistema.

Administrador: Persona encargada del mantenimiento y gestión de las bases de datos en la entidad. También puede encargarse de operar el sistema de réplica y el sistema propuesto como solución.

Sistema de Réplica: Aplicación de Sistema de réplica para bases de datos distribuidas en PostgreSQL existente en las FAR.

SIGNA: Sistema para la Sincronización Inicial y Gestión de Nodos Aislados para la replicación de bases de datos en PostgreSQL.

2.4 Especificación de los requerimientos del software.

2.4.1 Requerimientos funcionales.

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir sin tomar en consideración ningún tipo de restricción física. Se mantienen invariables sin importar con que propiedades o cualidades se relacionen. Definen el comportamiento interno del software: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas, que muestran cómo los casos de uso serán llevados a la práctica. A continuación se reflejan los requerimientos funcionales que han sido definidos en el sistema propuesto. [20]

RF1. Permitir conectar la aplicación a la BD.
RF2. Permitir configurar los datos.
RF2.1 Modificar fuente de datos.
RF2.2 Modificar destino de datos.
RF3. Permitir gestionar las reglas de los Destinos.
RF3.1. Agregar reglas en el Destino.
RF3.2. Modificar reglas en el Destino.
RF3.3. Eliminar reglas en el Destino.
RF4. Listar las BD destinos a sincronizar.
RF5. Sincronizar.
RF6. Gestionar nodo aislado sin conectividad.
RF6.1. Gestionar destino de datos (adicionar y eliminar).
RF7. Guardar sentencias SQL de la fuente de datos al destino configurado.
RF8. Cargar sentencias SQL en el destino de datos.
RF8.1. Confirmar ejecución de sentencias cargadas.
RF9. Guardar confirmación de ejecución de sentencias al destino de datos.
RF10. Cargar confirmación de ejecución de sentencias a la fuente de datos.
RF10.1. Eliminar las sentencias SQL confirmadas.
RF11. Mostar Reportes de Sincronización.
RF12. Mostrar Reportes sobre el proceso de la Gestión de los nodos aislados.

2.4.2 Requerimientos no funcionales.

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Características que hacen al producto atractivo, usable, rápido o confiable. En muchos casos son

fundamentales en el éxito del producto. Marcan la aceptación final por parte de los clientes y usuarios del producto. [21]

- **Apariencia o interfaz externa:**

  Interfaces de Usuarios: La interfaz debe ser amigable para el usuario para que se logre el adecuado entendimiento de la aplicación.

- **Usabilidad:**

  El software debe ser usado por personas autorizadas y capacitadas para el manejo de datos dentro de las FAR (Administradores de base de datos).

- **Rendimiento:**

  La de conexión a las BD debe ser de manera rápida y segura. El nivel de respuesta debe ser alto.

- **Hardware:**
  - **Para el cliente:**
  - Requerimientos mínimos: Procesador Pentium IV a 1.6GHz con 256 Mb de memoria RAM.

  Tarjeta de red, impresora.
  - **Para el servidor:**
  - Requerimientos mínimos: Procesador Pentium IV a 3GHz y 1Gb de memoria RAM.

  Al menos 40Gb de espacio libre en disco duro. Tarjeta de red.

  Nota: En los nodos aislados se podrá tener una PC Pentium IV a 1.6GHz con 256 Mb RAM y al menos 20Gb de espacio de HD que preste los servicios de servidor y cliente al mismo tiempo.

- **Software:**
  - **Para el cliente:** Sistema operativo GNU/Linux, cliente pgAdminIII, librería de interfaz gráfica GTK+, lenguaje de programación Python 2.4 ó superior y navegador web Mozilla Firefox.
  - **Para el servidor:** Sistema operativo GNU/Linux, gestor de base de datos PostgreSQL 8.0 o superior y lenguaje pgsql configurado.

- **Portabilidad:**

  El sistema está construido en código totalmente portable para sistema operativo GNU/Linux, plataforma usada por los servidores que existen actualmente en las FAR.

- **Seguridad:**

  La aplicación tiene un sistema de autentificación para administradores de las BD en donde se efectuarán las operaciones. Siendo estas personas las únicas con autorización para realizar las actividades dentro de sus respectivas bases de datos.

  - **Confiabilidad:**

    Alto nivel de confiabilidad. Conexión de forma segura a base de datos. El sistema debe avisar de los errores ocurridos mostrando sus respectivos reportes o de la efectiva realización de sus funciones.

  - **Integridad:**

    El transporte de datos entre los distintos nodos será de manera segura, de forma tal que los datos viajan de manera encriptados, asegurando su protección.

- **Ayuda y documentación en línea:**

  Se contará con un manual para la preparación de los usuarios que serán en este caso cada administrador de base de datos, donde se explicarán las características del sistema y sus funciones así como los detalles de su correcto uso. La interfaz contará con una ayuda en línea para guiar los distintos pasos de la aplicación.

- **Legales:**

  Este software será propiedad de las FAR, entregándose hasta el nivel de código fuente del sistema.

2.5 Descripción del sistema Propuesto.

2.5.1 Descripción de los Actores del Sistema.

Los actores del Sistema son aquellos terceros fuera del sistema que interactúan con él y pueden ser individuos u otros sistemas, por lo que a su vez son trabajadores del sistema y ejercen diferentes roles. [22]

En el sistema solo puede interactuar un actor que es el encargado de supervisar y gestionar sus funcionalidades, tal es el caso del Administrador de Base de Datos, que son los únicos autorizados a acceder y hacer cambios en sus respectivas Base de Datos.

Actores del Sistema	Descripción
Administrador de Base de Datos	Se encarga de Gestionar la Base de Datos con las funcionalidades que le brinda el sistema.

Tabla 1 Descripción de los Actores del Sistema.

2.5.2 Casos de Uso del Sistema.

Código	Caso de Uso
CU1	Conectar BD.
CU2	Configurar Datos.
CU3	Gestionar Reglas de los Destinos.
CU4	Sincronizar.
CU5	Gestionar Destino de Datos.
CU6	Cargar Confirmación.
CU7	Gestionar Fuente de Datos.
CU8	Cargar Sentencias.
CU9	Mostrar Reportes.

Tabla 2 Casos de Uso.

2.5.2.1 Diagrama de Casos de Uso del Sistema.

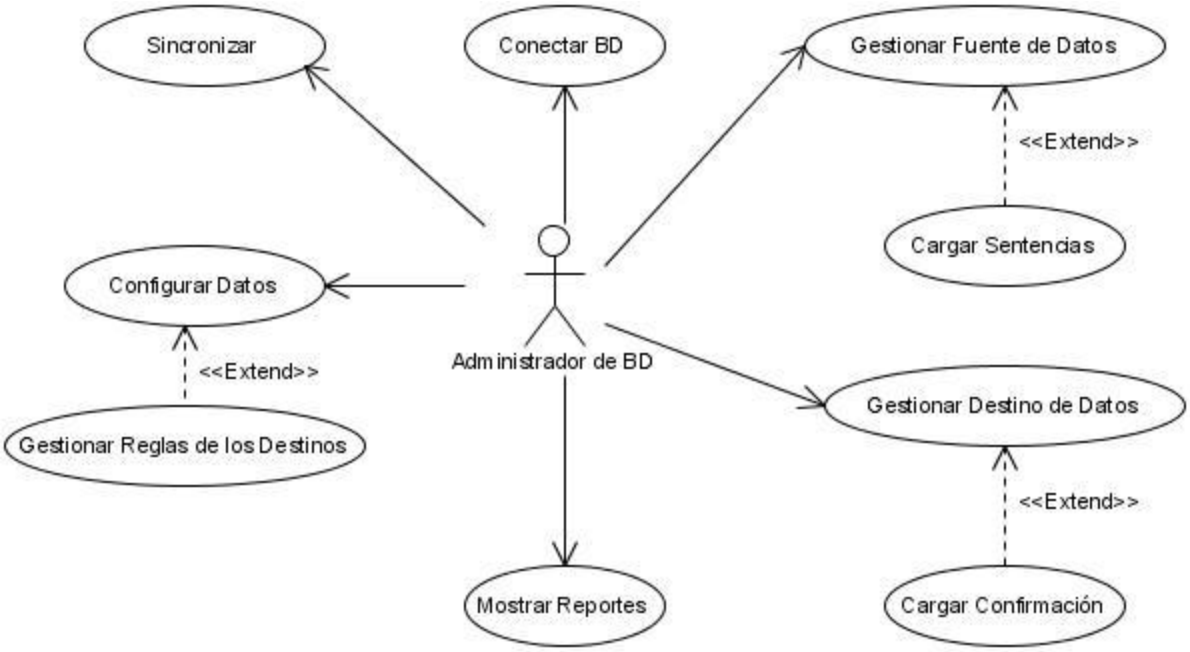


Fig. 8 Diagrama de Casos de Uso del Sistema.

2.5.2.2 Descripción de los Casos de Uso del Sistema.

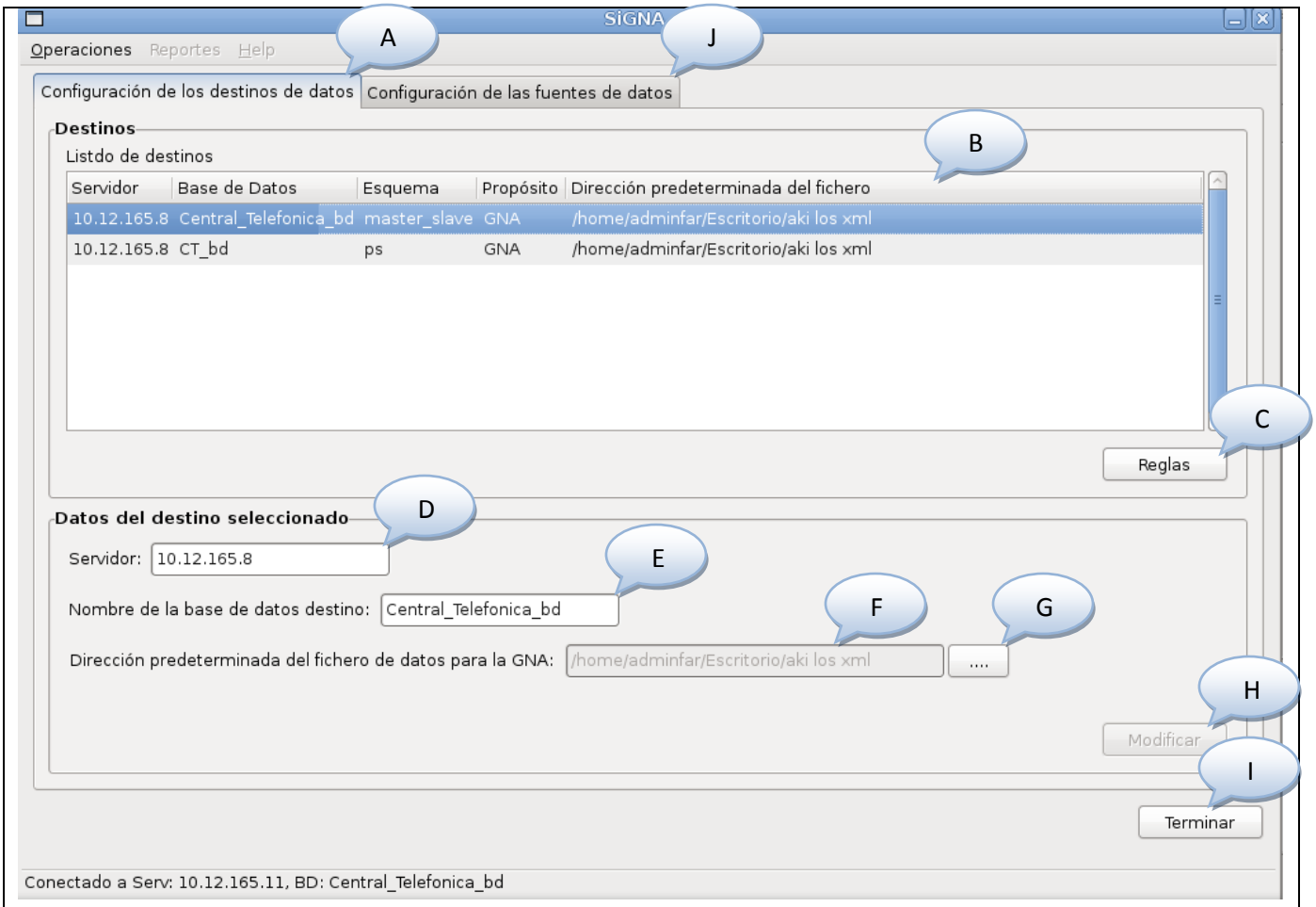
Caso de Uso	Nombre de Caso de Uso
CU1	Conectar BD
Actores	Administrador de Base de Datos.
Propósito	Conectar el sistema con la Base de Datos para realizar una acción.
Resumen	El caso de uso se inicia cuando el actor selecciona la opción de “Conectar BD” del menú principal y se muestra la interfaz de conexión. Culmina cuando el actor introduce los datos correctamente y seguidamente el sistema se conecta satisfactoriamente a la base de datos seleccionada.
Referencia	RF1
CU Relacionados	
Precondiciones	
Interfaz I	

Descripción.	
A: Entry para especificar número IP Servidor. B: Entry para especificar usuario. C: Entry para especificar contraseña. D: Entry para especificar número de Puerto. E: Button para conectarse al servidor. F: Combobox para seleccionar BD. G: Button para cancelar la acción de conexión. H: Button para aceptar la acción de conexión.	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El actor selecciona la opción “Conectar BD” del menú principal.	2. El sistema muestra la interfaz I para establecer la conexión a la BD.
3. El actor introduce los datos necesarios para conectarse con la BD: (Servidor, usuario, contraseña y puerto).	
4. El actor verifica si los datos son correctos y presiona el botón “Conectar”.	5. El sistema busca las bases de datos existentes y las lista en el combo “Seleccionar BD”.
6. El Actor selecciona una BD y presiona el botón “Aceptar”.	7. El sistema verifica que el esquema “SiGNA” existe en la BD y muestra la interfaz principal.

Cursos Alternos
Línea 5. Si los datos son incorrectos el sistema muestra un asistente de notificación de error detallando los campos con errores. Línea 6. Si el actor presiona el botón Cancelar el sistema muestra la interfaz principal en blanco. Línea 7. Si el esquema “SiGNA” no existe en la base de datos especificada, el sistema muestra un mensaje condicional dando la posibilidad de crear el esquema automáticamente.
Pos condiciones
• Queda habilitada la conexión a la base de datos. • Queda verificada la existencia del esquema “SiGNA” en la base de datos.
Puntos de Extensión.

Tabla 3 Descripción del Caso de Uso Conectar BD.

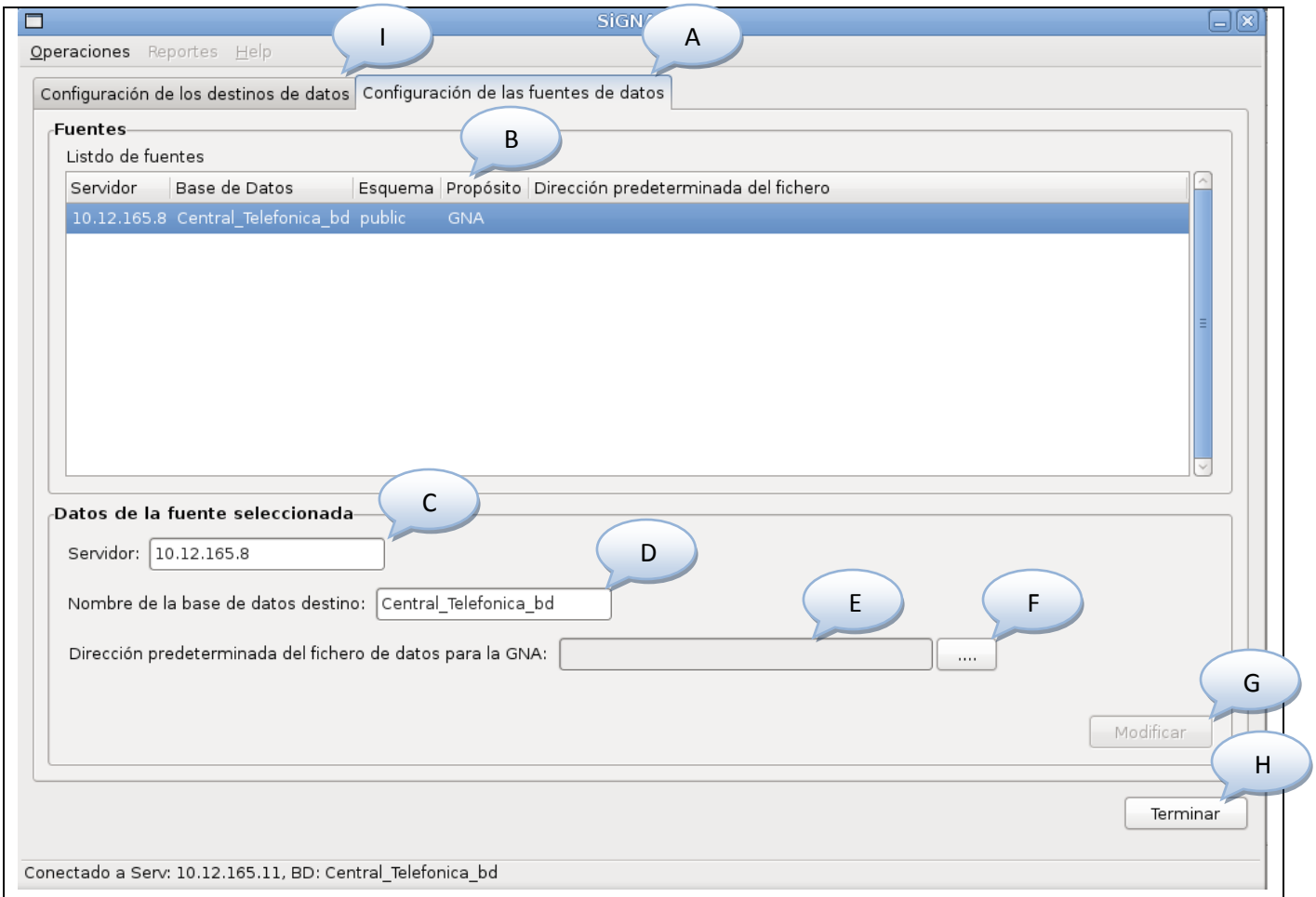
Caso de Uso	Nombre de Caso de Uso
CU2	Configurar Datos
Actores	Administrador de Base de Datos
Propósito	Establecer los datos de fuentes, destinos necesarios para la sincronización inicial y gestión de nodos sin conectividad.
Resumen	El caso de uso se inicia cuando el actor se conecta a la BD y escoge la opción “Configuración” del menú principal. El sistema muestra la interfaz de configuración. Concluye cuando el actor introduce los datos correctamente y el sistema actualiza los datos modificados.
Referencia	RF1, RF2, RF2.1, RF2.2
CU Relacionados.	CU Gestionar Reglas de los Destinos.
Precondiciones	El actor debe haberse conectado a la BD.
Interfaz I	



Descripción.
A: Pestaña para seleccionar configurar destinos de datos. B: TreeView para mostrar listado de destinos. C: Button para especificar reglas del destino seleccionado. D: Entry para especificar número IP servidor. E: Entry para especificar nombre de la BD destino. F: Entry para mostrar dirección predeterminada del fichero. G: Button para seleccionar dirección predeterminada del fichero. H: Button para realizar la modificación. I: Button terminar la operación de configuración. J: Pestaña para seleccionar configurar fuente de datos.
Flujo Normal de Eventos

Acción del Actor	Respuesta del Sistema
1. El actor escoge la opción “Configuración”	2. El sistema muestra la interfaz I con un

del menú principal.	listado de destinos configurados para esa fuente.
3. El actor escoge: • La pestaña “Configuración de los destinos de datos”. • La pestaña “Fuentes”. Ver sección: Modificar Fuentes.	4. El sistema muestra un listado con los destinos existentes en la base de datos.
5. El actor escoge un destino del listado de destinos.	6. El sistema muestra un grupo de campos para editar los datos del destino seleccionado.
7. El actor presiona el botón “…”.	8. Se muestra un dialogo de búsqueda para seleccionar el directorio donde se va a crear el fichero.
9. El actor presiona el botón “Aceptar” del dialogo de búsqueda.	10. El sistema muestra en el Entry F la dirección predeterminada del fichero de datos.
11. El actor introduce los datos del servidor y el nombre de la BD destino, verifica si son correctos y presiona • el botón “Modificar” • el botón “Reglas”. Ver CU Gestionar Reglas de los Destinos	12. El sistema actualiza los datos en el listado de destinos.
Cursos Alternos	
Línea 9. Si el actor presiona el botón “Cancelar” del dialogo de búsqueda el sistema cancela la operación de configuración y se repite el flujo normal de eventos desde la línea 2. Línea 11. Si el actor presiona el botón “Terminar” se descartan los cambios realizados en el destino seleccionado. Línea 12. Si los datos entrados son incorrectos el sistema muestra un mensaje de error con la descripción detallada del campo que contiene los datos incorrectos.	
Sección: Modificar Fuentes	
Interfaz II	



Descripción.
A: Pestaña para seleccionar configurar fuente de datos. B: TreeView para mostrar listado de fuentes. C: Entry para especificar número IP servidor. D: Entry para especificar nombre de la BD fuente. E: Entry para mostrar dirección predeterminada del fichero. F: Button para seleccionar dirección predeterminada del fichero. G: Button para realizar la modificación. H: Button terminar la operación de configuración. I: Pestaña para seleccionar configurar destino de datos.

Acción del Actor	Respuesta del Sistema
1. El actor escoge la pestaña “Configuración de las fuentes de datos”.	2. El sistema muestra la interfaz II con un listado de fuente configuradas para ese destino.

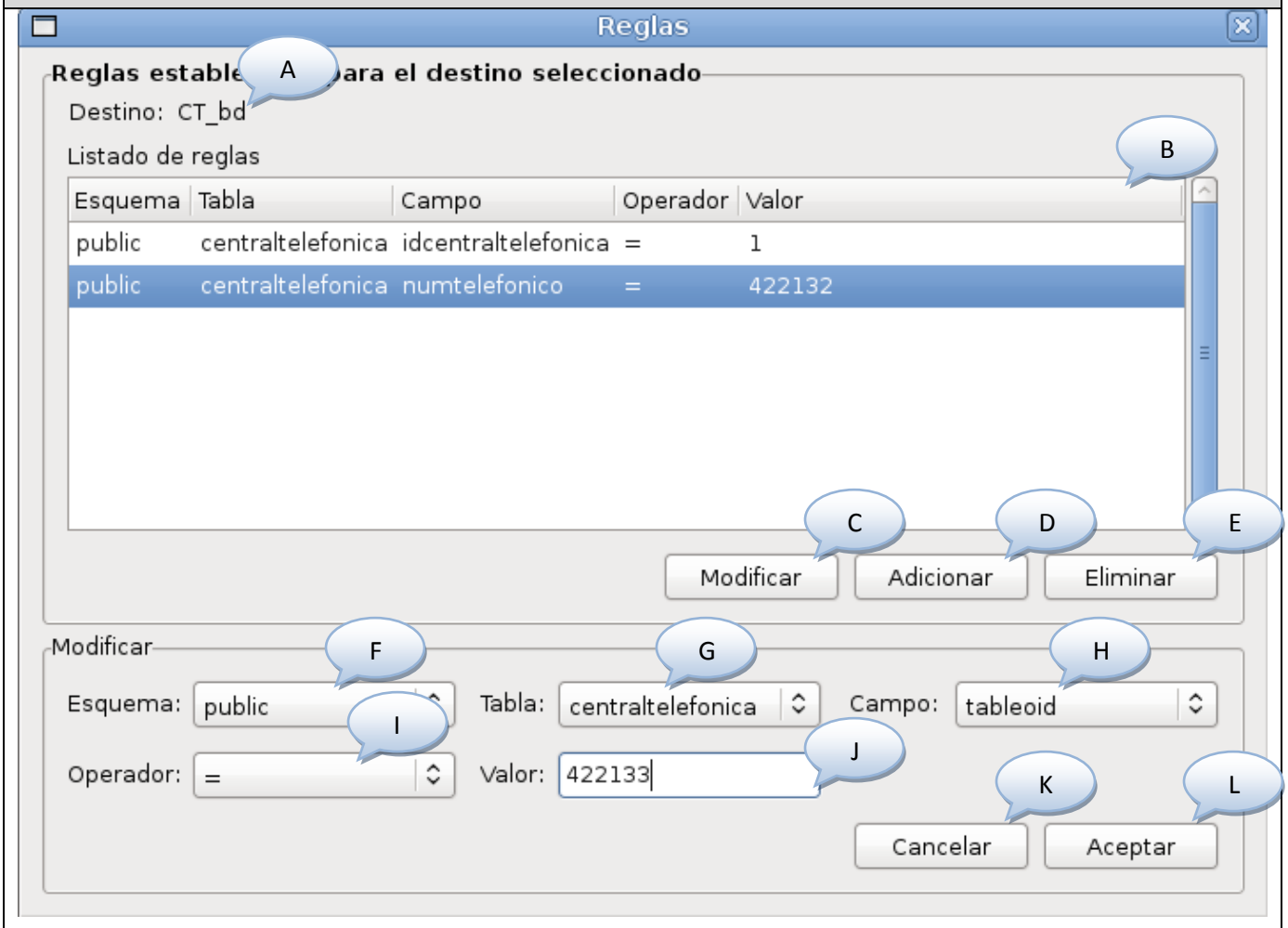
3. El actor selecciona una fuente del listado de fuentes.	4. El sistema muestra un grupo de campos para editar los datos de la fuente seleccionada.
5. El actor presiona el botón "…".	6. Se muestra un dialogo de búsqueda para seleccionar el directorio donde se va a crear el fichero.
7. El actor presiona el botón "Aceptar" del dialogo de búsqueda.	8. El sistema muestra en el Entry E la dirección predeterminada del fichero de datos.
9. El actor verifica que los datos sean correctos y presiona el botón "Modificar".	10. El sistema modifica los datos de la fuente que fue seleccionada anteriormente.
11. El actor presiona el botón "Terminar".	12. El sistema termina la operación de configuración de fuentes de datos.
Cursos Alternos	
Línea 5. Si el actor presiona el botón "Terminar" se descartan los cambios realizados en la fuente seleccionada. Línea 7. Si el actor presiona el botón "Cancelar" del dialogo de búsqueda, el sistema cancela la operación de configuración y se repite el flujo normal de eventos desde la línea 2. Línea 9. Si el actor presiona el botón "Terminar", se descartan los cambios realizados en la fuente seleccionada y muestra la interfaz principal. Línea 10. Si los datos entrados son incorrectos el sistema muestra un mensaje de error con la descripción detallada del campo que contiene los datos incorrectos.	
Pos condiciones.	
Quedan configurados los destinos y las fuentes en la BD.	
Puntos de Extensión.	
Línea 7 del Flujo normal de eventos • El actor presiona el botón "Reglas". Ver CU Gestionar Reglas de los Destinos.	

Tabla 4 Descripción del Caso de Uso Configurar Datos.

Caso de Uso	Nombre de Caso de Uso
CU3	Gestionar Reglas de los Destinos
Actores	Administrador de Base de Datos

Propósito	Establecer las reglas para cada destino configurado.
Resumen	El Caso de uso de inicia para dar respuesta a la solicitud del CU Configurar Datos. Culmina cuando el actor presiona el botón "Aceptar" luego de haber verificado si los datos entrados son correctos y seguidamente el sistema actualiza los datos de las reglas contenidas en el destino.
Referencia	RF1, RF3, RF3.1, RF3.2, RF3.3.
CU Relacionados.	
Precondiciones	El actor debe haberse conectado a la BD.

Interfaz I



Descripción:

A: Label que muestra el nombre del destino seleccionado.

B: TreeView que muestra el listado de reglas configuradas para el destino seleccionado.

C: Button para modificar la regla seleccionada.

D: Button para adicionar una nueva regla.

E: Button para eliminar la regla seleccionada.

F: Combobox para modificar el nombre del esquema de la regla seleccionada.

G: Combobox para modificar el nombre de la tabla de la regla seleccionada.

H: Combobox para modificar el nombre del campo de la regla seleccionada.

I: Combobox para modificar el tipo de operador de la regla seleccionada.

J: Entry para modificar el valor de la regla seleccionada.

K: Button para cancelar la operación de gestionar reglas.

L: Button para aceptar la operación de gestionar reglas.

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El actor presiona el botón “Reglas” de la interfaz “Configuración de los destinos de datos”.	2. El sistema muestra la interfaz I con un listado de reglas configuradas para ese destino.
3. El actor selecciona una regla del listado y presiona el botón “Modificar”.	4. El sistema muestra los campos para editar los parámetros de la regla.
5. El actor introduce los datos en los campos, verifica si son correctos y presiona el botón “Aceptar”	6. El sistema actualiza el listado de reglas contenidas en el destino seleccionado.

Cursos Alternos

Línea 3.

- El actor presiona el botón “Adicionar”, el sistema limpia los campos con los datos introducidos y continúa el flujo normal de eventos.
- El actor presiona el botón “Eliminar” y el sistema elimina la regla seleccionada.

Línea 5. Si el actor presiona el botón “Cancelar”, el sistema descarta los cambios realizados en las reglas del destino seleccionado.

Línea 6. Si los datos entrados son incorrectos el sistema muestra un mensaje de error con la descripción detallada del campo que contiene los datos incorrectos.

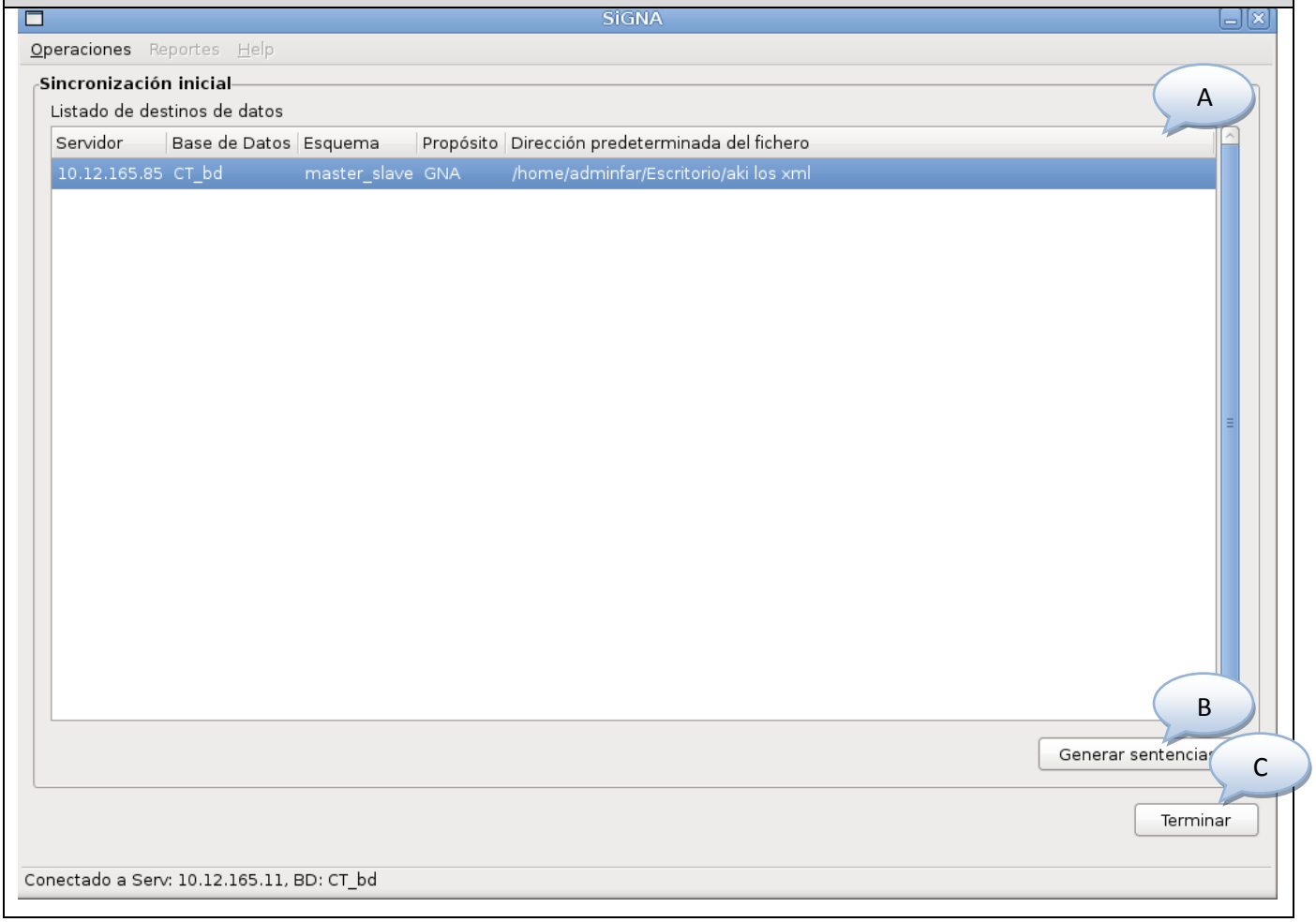
Pos condiciones

Quedan gestionadas las reglas de los destinos en la base de datos.

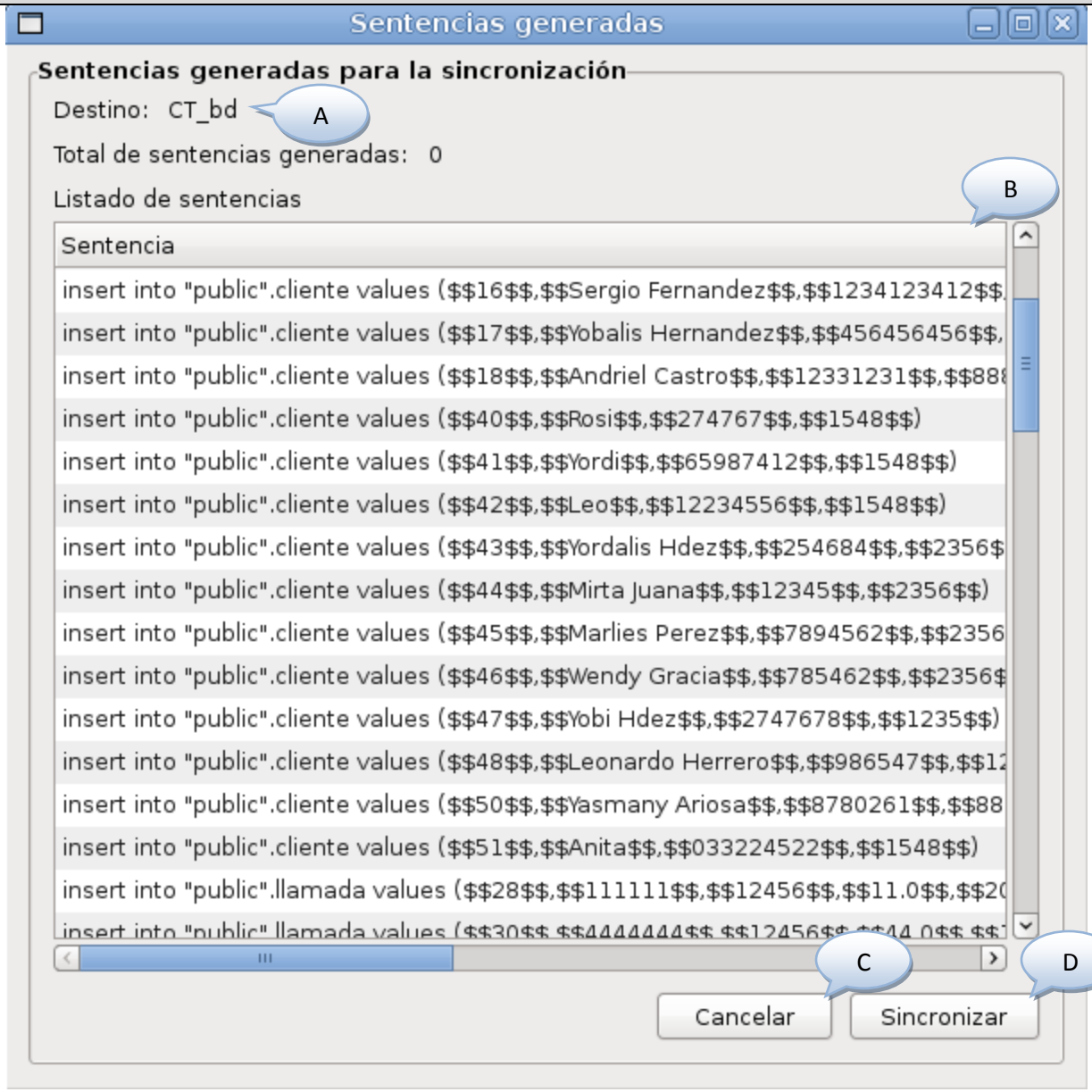
Puntos de Extensión.

Tabla 5 Descripción del Caso de Uso Gestionar Reglas de los Destinos.

Caso de Uso	Nombre de Caso de Uso
CU4	Sincronizar.
Actores	Administrador de Base de Datos
Propósito	Realizar la sincronización inicial en un destino seleccionado.
Resumen	Este caso de uso se inicia cuando el actor selecciona la opción de "Sincronizar" en el menú principal. Se muestra la interfaz de Sincronización y culmina cuando el actor presiona el botón de terminar y el sistema culmina la acción de sincronización.
Referencia	RF1, RF4, RF5
CU Relacionados	
Precondiciones	El actor debe haberse conectado a la BD.
Interfaz I	



Descripción.
A: TreeView que muestra el listado de destino configurados para la sincronización. B: Button para generar las sentencias SQL. C: Button para terminar el proceso de sincronización.
Interfaz II



Descripción
A: Label que muestra el nombre de la BD seleccionada. B: TreeView que muestra el listado de sentencias generadas. C: Button que cancela el proceso de sincronización.

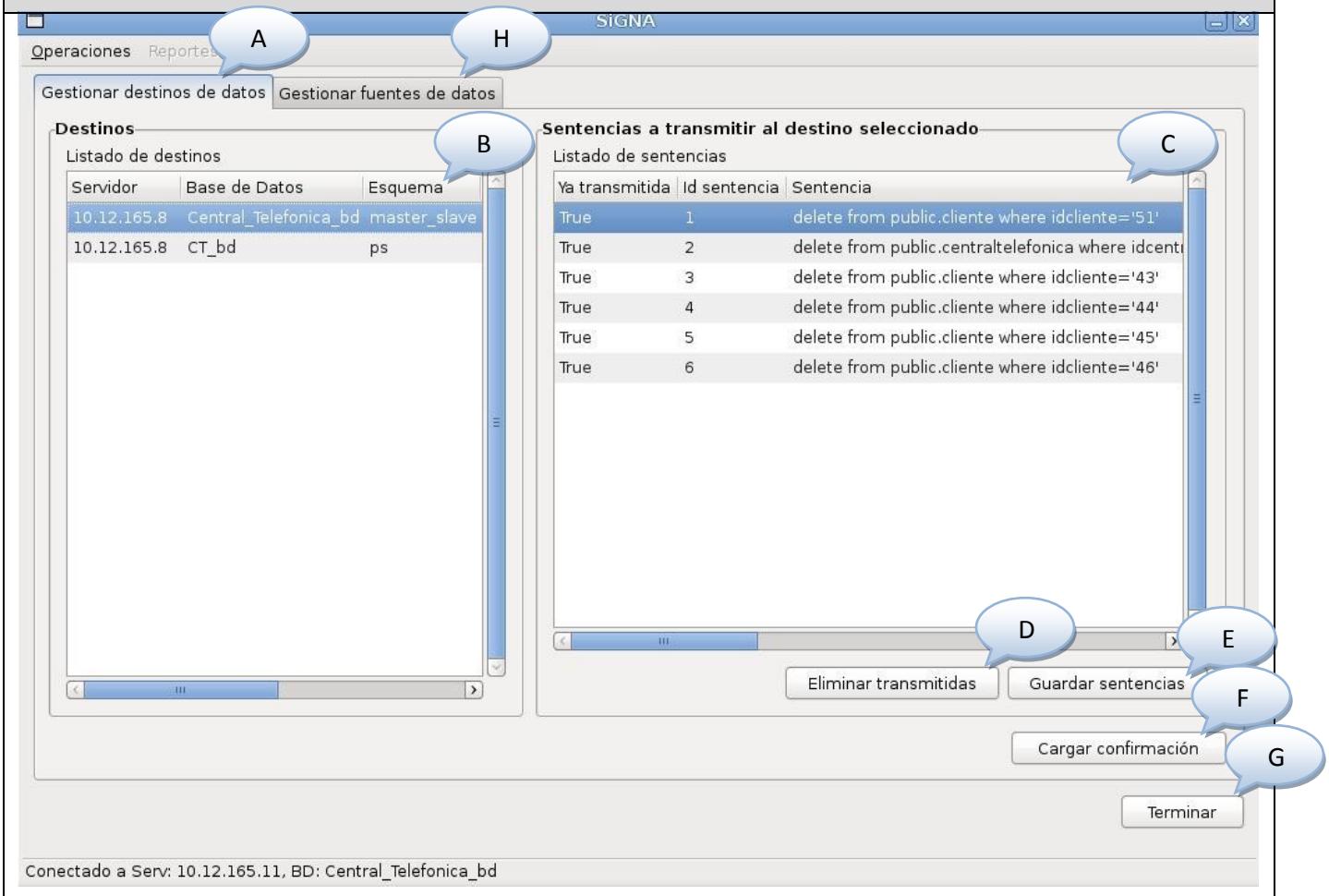
D: Button realiza el proceso de sincronización.	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El actor selecciona la opción de Sincronizar en el menú principal.	2. El sistema muestra la interfaz I con un listado de destinos de datos configurados para realizar la sincronización.
3. El actor selecciona un destino y presiona el botón “Generar Sentencias”.	4. El sistema muestra la interfaz II con las sentencias generadas para ese destino seleccionado.
5. El actor presiona el botón “Sincronizar”.	6. El sistema guarda las sentencias en la tabla de transmisión y muestra la interfaz I.
7. El actor presiona el botón “Terminar”.	8. El sistema culmina la acción de sincronización y muestra la interfaz principal.
Cursos Alternos	
Línea 3. Si el actor presiona el botón “Terminar” no se ejecuta ninguna acción y el sistema muestra la interfaz principal. Línea 5. Si el actor presiona el botón “Cancelar” el sistema muestra la interfaz I y se repite el flujo normal de eventos desde la línea 3.	
Pos condiciones	
Queda realizada la sincronización inicial para el destino seleccionado.	
Puntos de Extensión.	

Tabla 6 Descripción del Caso de Uso Sincronizar.

Caso de Uso	Nombre de Caso de Uso
CU5	Gestionar Destino de Datos.
Actores	Administrador de Base de Datos
Propósito	Gestionar los destinos de datos para una BD fuente, así como las confirmaciones con los ID de las sentencias SQL provenientes de una BD destino, aislada totalmente por la red, utilizando vías manuales.

Resumen	El Caso de uso se inicia cuando el actor elige la opción “Gestionar Nodos Aislados” en el menú principal, el sistema muestra la interfaz de gestionar nodos aislados, luego el actor selecciona la pestaña “Gestionar destino de datos” y el sistema muestra la interfaz de gestión de destino de datos. Culmina cuando el actor presiona el botón terminar, el sistema finaliza la acción de gestionar destinos de datos y muestra la interfaz principal.
Referencia	RF1, RF6, RF7.
CU Relacionados	CU Cargar Confirmación.
Precondiciones	El actor debe haberse conectado a la BD.

Interfaz I



Descripción:

A: Pestaña para seleccionar Gestionar destinos de datos.

B: TreeView que muestra el listado de destinos configurados para la gestión de nodos aislados.

C: TreeView que muestra el listado de sentencias del destino seleccionado.

D: Button para eliminar las sentencias que fueron transmitidas.

E: Button para guardar las sentencias en el fichero de transmisión.

F: Button para cargar las confirmaciones del fichero de confirmación.

G: Button para terminar la acción de gestionar destinos de datos.

H: Pestaña para seleccionar Gestionar fuentes de datos.

Flujo Normal de Eventos

Acción del Actor	Respuesta del Sistema
1. El actor selecciona la opción “Nodos Aislados” en el menú principal.	2. El sistema muestra la interfaz I para gestionar los nodos aislados.
3. El actor Escoge la pestaña “Gestionar destinos de datos”.	4. El sistema muestra un listado de destinos configurados para el tipo Nodos Aislados.
5. El actor selecciona un destino del listado de destinos.	6. El sistema muestra un listado de las sentencias generadas para el destino seleccionado.
7. El actor presiona el botón: • “Guardar Sentencias”. • “Cargar confirmación”. Ver CU Cargar Confirmación.	8. El sistema muestra un diálogo de búsqueda para guardar el fichero.
9. El actor selecciona el lugar donde va a guardar el fichero y presiona el botón “Aceptar”	10. El sistema guarda los datos en el fichero y lo guarda en la posición escogida.
11. El actor presiona el botón “Terminar”	12. El sistema culmina la acción de gestionar destinos de datos y muestra la interfaz principal.

Cursos Alternos

Línea 7.

- Si el actor presiona el botón “Eliminar Transmitidas” el sistema elimina las sentencias que fueron transmitidas y se repite el flujo normal de eventos desde la línea 4.
- Si el actor presiona el botón “Terminar” el sistema no guarda las sentencias, culmina la acción de gestionar destinos de datos y muestra la interfaz principal.

Línea 9.

- Si el actor presiona el botón “Cancelar” del dialogo de búsqueda el sistema no guarda las

sentencias y se repite el flujo normal de eventos desde la línea 6.
Pos condiciones
• Queda creado el fichero XML en la dirección seleccionada por el actor. • Quedan guardadas las sentencias SQL en el fichero XML, listas a ser transmitidas por vía manual a la BD Destino.
Puntos de Extensión.
Línea 7 del flujo normal de eventos • El actor presiona el botón “Cargar confirmación”. Ver CU Cargar Confirmación.

Tabla 7 Descripción del Caso de Uso Gestionar Destino de Datos.

Caso de Uso	Nombre de Caso de Uso
CU6	Cargar Confirmación.
Actores	Administrador de Base de Datos
Propósito	Cargar confirmaciones de un fichero XML procedente de una BD Fuente localizada en un nodo totalmente aislado por la red, trasladado por vías manuales.
Resumen	El Caso de uso se inicia para dar respuesta a la solicitud del CU Gestionar Destino de Datos. El sistema muestra un diálogo de búsqueda para cargar el fichero XML de confirmación. Culmina cuando el actor acepta la confirmación y el sistema actualiza los datos del destino confirmado.
Referencia	RF1, RF6, RF10, RF10.1.
CU Relacionados	
Precondiciones	El actor debe haberse conectado a la BD.
Interfaz I	

Descripción.

A: Label que muestra la fecha en que fue creado el fichero de confirmación.

B: Label que muestra el nombre del esquema de recepción de datos.

C: Label que muestra el IP del servidor de la fuente de datos.

D: Label que muestra el nombre de la BD de la fuente de datos.

E: Label que muestra el IP del servidor del destino de datos.

F: Label que muestra el nombre de la BD del destino de datos.

G: TreeView que muestra el listado de confirmaciones con sus sentencias correspondientes.

H: Button para recargar las confirmaciones en el fichero de acuse.

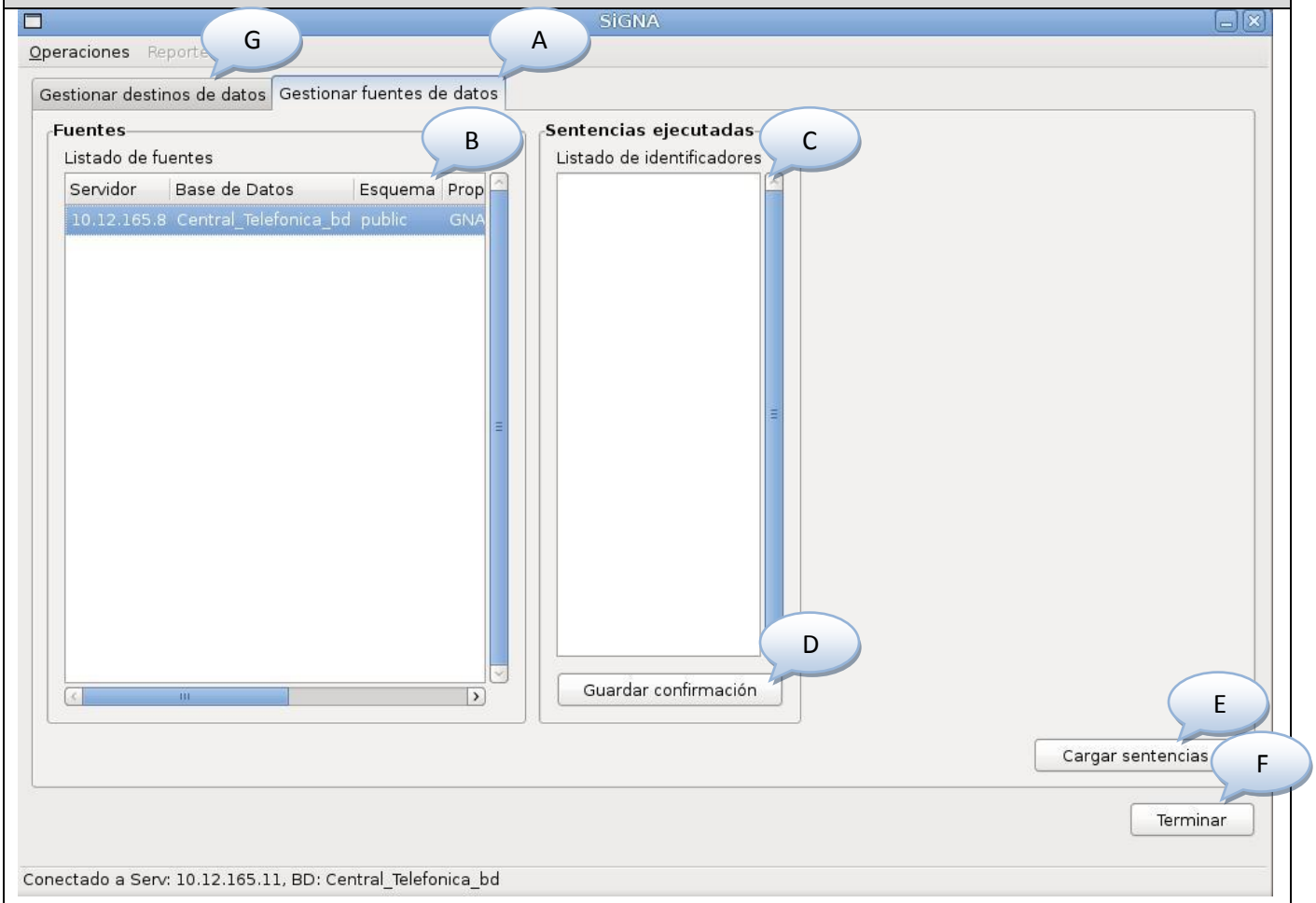
I: Button para ejecutar las confirmaciones en la BD Fuente.	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
	1. El sistema muestra un diálogo de búsqueda para cargar el fichero XML de confirmación.
2. El actor selecciona el fichero y presiona el botón “Aceptar”.	3. El sistema muestra la Interfaz I con la información del fichero y un listado de confirmaciones proveniente del fichero.
4. El actor presiona el botón “Aceptar Confirmación”.	5. El sistema verifica si la información de la fuente de datos coincide con el servidor local y actualiza los datos para el destino confirmado y muestra la interfaz “Gestionar destinos de datos”.
Cursos Alternos	
Línea 2. Si el actor presiona el botón “Cancelar” el sistema no realiza ninguna acción y muestra la interfaz “Gestionar destinos de datos”. Línea 4. Si el actor presiona el botón “Recargar” el sistema carga de nuevo las sentencias que se encuentran en el fichero de transmisión. Línea 5. Si la información no coincide el sistema muestra un mensaje con la información detallada del error.	
Pos condiciones	
• Quedan cargadas las confirmaciones procedentes de la BD Destino. • Quedan actualizados los datos del destino perteneciente al fichero de confirmación.	
Puntos de Extensión.	

Tabla 8 Descripción del Caso de Uso Cargar Confirmación.

Caso de Uso	Nombre de Caso de Uso
CU7	Gestionar Fuente de Datos.
Actores	Administrador de Base de Datos
Propósito	Gestionar las fuentes de datos para una BD destino, así como las sentencias SQL provenientes de una BD destino aislado totalmente por la red, utilizando vías manuales.

Resumen	El Caso de uso se inicia cuando el actor elige la opción “Nodos Aislados” en el menú principal. El sistema muestra la interfaz de Nodos Aislados. El actor escoge la pestaña “Destinos” y presiona el botón “Guardar Confirmación”. Culmina cuando el actor presiona el botón de “Terminar” y el sistema finaliza la acción de gestión de fuentes de datos.
Referencia	RF1, RF6, RF9.
CU Relacionados.	CU Cargar Sentencias.
Precondiciones	El actor debe haberse conectado a la BD.

Interfaz I



Descripción.

A: Pestaña para seleccionar Gestionar fuente de datos.

B: TreeView que muestra el listado de Fuentes configuradas para el destino.

C: TreeView que muestra el listado de identificadores de las sentencias que han sido ejecutadas en el destino para la fuente seleccionada.

D: Button para guardar las confirmaciones en el fichero de acuse.

E: Button para cargar las sentencias SQL del fichero de transmisión.

F: Button para terminar la acción de Gestionar las fuentes de datos.

G: Pestaña para seleccionar Gestionar destino de datos

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El actor elige la opción “Nodos Aislados” en el menú principal.	2. El sistema muestra la interfaz de Nodos Aislados.
3. El actor escoge la pestaña “Gestionar fuentes de datos”.	4. El sistema muestra la Interfaz I con un listado de fuentes configuradas para ese destino.
5. El actor selecciona una fuente del listado.	6. El sistema muestra los identificadores de las confirmaciones para esa fuente.
7. El actor presiona el botón • “Guardar Confirmación” • “Cargar sentencias”. Ver Cu Cargar Sentencias.	8. El sistema muestra un dialogo de búsqueda para guardar el fichero XML.
9. El actor selecciona la dirección y presiona el botón “Aceptar”	10. El sistema crea y guarda el fichero, seguidamente actualiza el listado de identificadores de la fuente.
11. El actor presiona el botón “Terminar”.	12. El sistema culmina la acción de gestionar las fuentes de datos y muestra la interfaz principal.

Cursos Alternos

Línea 7.

- Si el actor presiona el botón “Terminar” el sistema culmina la acción de gestionar las fuentes de datos y muestra la interfaz principal.

Línea 9.

- Si el actor presiona el botón “Cancelar” el sistema no crea el fichero de confirmación, no se realiza ningún cambio en la base de datos y se repite el flujo normal de eventos desde la línea 4.

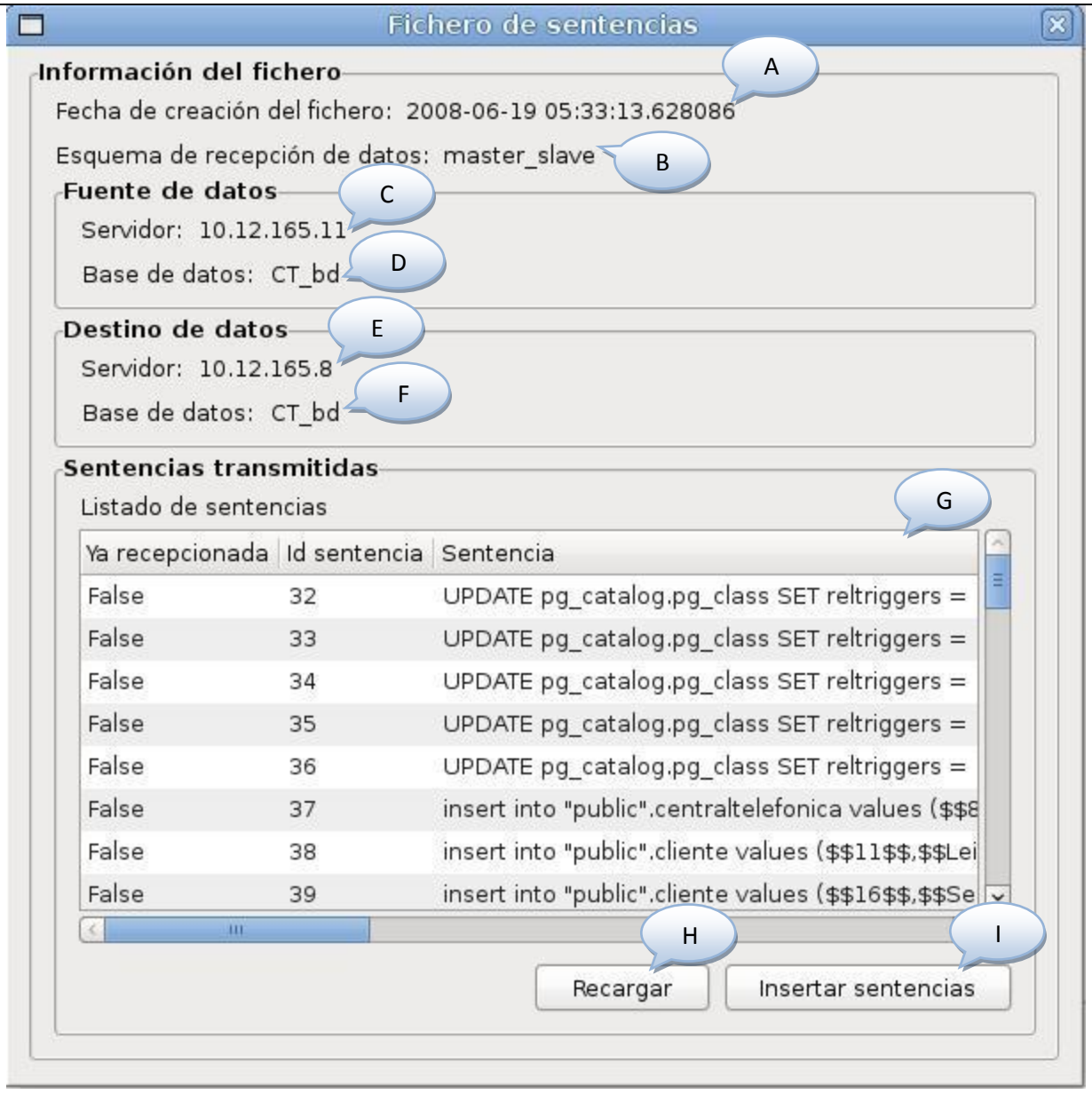
Pos condiciones

- Queda creado el fichero XML de confirmación.
- Quedan guardadas las confirmaciones en el fichero, listas para su transmisión manual hacia en

la BD fuente.
Puntos de Extensión.
Línea 7 del Flujo Normal de Eventos • El actor presiona el botón “Cargar sentencias”. Ver CU Cargar Sentencias.

Tabla 9 Descripción del Caso de Uso Gestionar Fuente de Datos.

Caso de Uso	Nombre de Caso de Uso
CU8	Cargar Sentencias.
Actores	Administrador de Base de Datos
Propósito	Cargar las sentencias procedentes de una BD localizada en un nodo totalmente aislado por la red a través de un fichero de transmisión XML, utilizando vías manuales.
Resumen	El caso de uso se inicia para dar respuesta a la solicitud del CU Gestionar Fuente de Datos. El sistema muestra un dialogo de búsqueda para buscar el fichero de transmisión de sentencias. Culmina cuando el actor presiona el botón “Insertar sentencias” y el sistema ejecuta las sentencias SQL en la BD.
Referencia	RF1, RF6, RF6.1, RF8, RF8.1.
CU Relacionados	
Precondiciones	El actor debe haberse conectado a la BD.
Interfaz I	



Descripción.

A: Label que muestra la fecha de creación del fichero de transmisión.

B: Label que muestra el nombre del esquema de recepción de los datos.

C: Label que muestra el número IP del servidor de la fuente de datos.

D: Label que muestra el nombre de la BD fuente.

E: Label que muestra el número IP del servidor del destino de datos.

F: Label que muestra el nombre de la BD destino.

G: TreeView que muestra el listado de sentencias contenidas en el fichero de transmisión.

H: Button que recarga los datos de las sentencias contenidas en el fichero de transmisión.

<table>
<tr><td colspan="2">I: Button inserta en la BD las sentencias listadas.</td></tr>
<tr><td colspan="2">Flujo Normal de Eventos</td></tr>
<tr><td>Acción del Actor</td><td>Respuesta del Sistema</td></tr>
<tr><td></td><td>1. El sistema muestra un dialogo de búsqueda para buscar el fichero de transmisión de sentencias.</td></tr>
<tr><td>2. El actor selecciona el fichero y presiona el botón “Aceptar”</td><td>3. El sistema muestra la interfaz I con la información del fichero y una lista de sentencias provenientes del fichero.</td></tr>
<tr><td>4. El actor presiona el botón “Insertar Sentencias”</td><td>5. El sistema verifica si la información del destino de datos coincide con el servidor local y ejecuta las sentencias en la BD y muestra la interfaz de Gestionar fuentes de datos.</td></tr>
<tr><td colspan="2">Cursos Alternos</td></tr>
<tr><td colspan="2">Línea 2. Si el actor presiona el botón “Cancelar” el sistema no realiza ninguna acción y muestra la interfaz de Gestionar fuentes de datos.
Línea 4. Si el actor presiona el botón “Recargar” el sistema recarga las sentencias que se encuentran en el fichero de transmisión.
Línea 5.
• Si la información no coincide el sistema muestra un mensaje con la información detallada del error.
• La información del destino coincide y no existe el esquema de recepción de datos en la BD, el sistema muestra un mensaje condicional permitiendo crear el esquema inexistente.</td></tr>
<tr><td colspan="2">Pos condiciones</td></tr>
<tr><td colspan="2">Quedan ejecutadas las sentencias SQL en la BD destino.</td></tr>
<tr><td colspan="2">Puntos de Extensión.</td></tr>
<tr><td colspan="2"></td></tr>
</table>

Tabla 10 Descripción del Caso de Uso Cargar Sentencias.

Caso de Uso	Nombre de Caso de Uso
CU9	Mostrar Reportes.
Actores	Administrador de Base de Datos
Propósito	Mostrar distintos tipos de reportes para controlar y verificar el funcionamiento

	del sistema así como los datos transmitidos.
Resumen	Este Caso de uso se inicia cuando el actor selecciona la opción “Mostrar Reporte” en el menú principal. El sistema permite mostrar cinco tipos de reportes: Por sincronizaciones realizadas, sentencias guardadas, sentencias cargadas, confirmaciones guardadas y confirmaciones cargadas. Culmina cuando el sistema muestra el reporte del tipo seleccionado por el actor.
Referencia	RF1, RF11, RF12.
CU Relacionados	
Precondiciones	El actor debe haberse conectado a la BD.
Interfaz I	

Descripción.
A: ComboBox para seleccionar el tipo de reporte.
B: ComboBox para seleccionar el nombre de la base de datos.
C: CheckBox para activar el filtrado por fecha de creada la sentencia.
D: Entry para especificar la fecha de inicio.
E: Entry para especificar la fecha fin.
F: CheckBox para activar el filtrado por fecha de reportada la sentencia.
G: Entry para especificar la fecha de inicio.
H: Entry para especificar la fecha fin.
I: CheckBox para activar el filtrado por sentencias con error.

<table>
<tr><td colspan="2">J: Button para mostrar el reporte.</td></tr>
<tr><td colspan="2">Flujo Normal de Eventos</td></tr>
<tr><td>Acción del Actor</td><td>Respuesta del Sistema</td></tr>
<tr><td>1. El actor selecciona la opción “Reporte” en el menú principal.</td><td>2. El sistema muestra la Interfaz I con los campos para seleccionar los datos del reporte.</td></tr>
<tr><td>3. El actor selecciona un tipo de reporte en el combobox “Tipo de reporte”.</td><td>4. El sistema carga y muestra las bases de datos en el combobox “Base de datos” de acuerdo al tipo de reporte seleccionado y se activan los campos para filtrar el reporte.</td></tr>
<tr><td>5. El actor selecciona la base de datos para filtrar el reporte.</td><td></td></tr>
<tr><td>6. El actor marca el checkbox “Fecha de creadas las sentencias”.</td><td>7. El sistema activa los campos para especificar el intervalo de fechas de creadas las sentencias.</td></tr>
<tr><td>8. El actor especifica las fechas de inicio y fin para el intervalo de creación de las sentencias.</td><td></td></tr>
<tr><td>9. El actor marca el checkbox “Fecha de reportada las sentencias”.</td><td>10. El sistema activa los campos para especificar el intervalo de fechas de reportadas las sentencias.</td></tr>
<tr><td>11. El actor especifica las fechas de inicio y fin para el intervalo de reportadas las sentencias.</td><td></td></tr>
<tr><td>12. El actor marca el checkbox “Sentencias con error”.</td><td></td></tr>
<tr><td>13. El actor verifica si los datos son correctos y presiona el botón reportar.</td><td>14. El sistema carga y muestra el reporte filtrado en el navegador web Mozilla Firefox.</td></tr>
<tr><td colspan="2">Cursos Alternos</td></tr>
<tr><td colspan="2">Línea 4
• Si no existen bases de datos disponibles para el tipo de reporte seleccionado, el sistema no activa los campos para filtrar el reporte y se repite el flujo normal de eventos desde la línea 2.
Línea 5</td></tr>
</table>

• Si el actor no selecciona la base de datos, el sistema no filtra el reporte por una base de datos específica. Línea 6 • Si el actor no marca el checkbox “Fecha de creadas las sentencias”, el sistema no filtra el reporte por fecha de creación de las sentencias. Línea 9 • Si el actor no marca el checkbox “Fecha de reportada las sentencias”, el sistema no filtra el reporte por la fecha de reportadas las sentencias. Línea 12 • Si el actor no marca el checkbox “Sentencias con error”, el sistema no filtra el reporte por sentencias con error. Línea 14 • Si los datos especificados no son correctos, el sistema muestra un mensaje de error con los campos incorrectos.
Pos condiciones
Queda mostrado el reporte para ser analizado.
Puntos de Extensión.

Tabla 11 Descripción del Caso de Uso Mostrar Reportes.

2.6 Conclusiones

En este capítulo se ha abordado sobre la elaboración y uso de un Modelo de Dominio, materializando los principales conceptos, entidades y requerimientos del sistema. De esta manera queda delimitada una clara visión del nivel de respuesta deseado y exigido para su posterior modelación, análisis y diseño. También se han presentados los actores que interactuarán de alguna medida con el sistema a automatizar y una pequeña descripción de cada uno, así como un diagrama dejando ver las relaciones que entre ellos existen. En el caso del propio sistema se observa que solo se cuenta con un actor, pues está dirigida a las personas que manejan las distintas BD. También se hace una descripción detallada de cada uno de los casos de uso, demostrando así las condiciones que tendrá el sistema propuesto.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DE LA SOLUCIÓN PROPUESTA.

3.1 Introducción.

En el siguiente capítulo se tratan de comprender perfectamente los requisitos del sistema mediante la modelación de las clases del análisis y se precisará como se implementará la solución a través de la modelación de las clases del diseño. Además de esto, se presentan los diagramas de interacción, se expone el modelo de datos a partir de de las clases persistentes detectadas en el diseño y los principios de diseño, concepción general de la ayuda, el tratamiento de errores, entre otros.

3.2 Análisis.

3.2.1 Diagrama de clases del análisis.

El diagrama de clases del análisis es un artefacto en el que se representan los conceptos del dominio del problema, como parte del mundo real, no desde el punto de vista de la construcción del sistema. [23]

3.2.1.1 Diagrama de clases del análisis por Casos de Uso.

Fig. 9 Diagrama de Clases del Análisis CU Conectar BD.

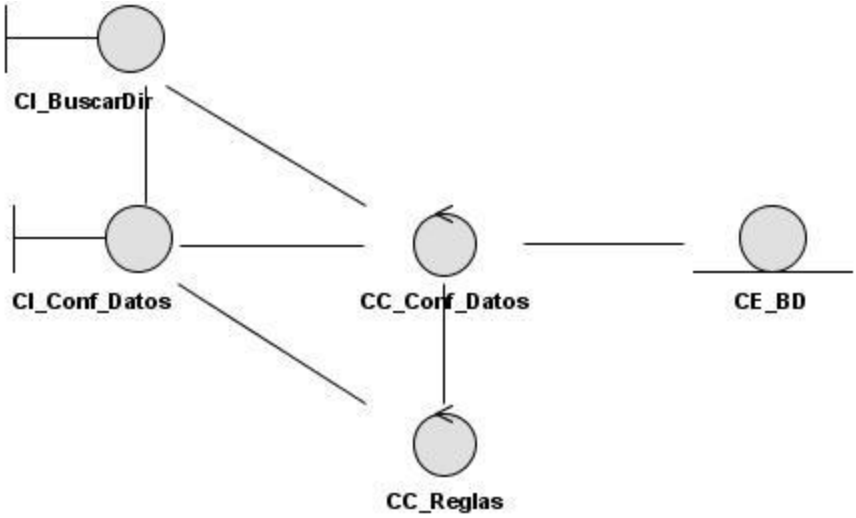


Fig. 10 Diagrama de Clases del Análisis CU Configurar Datos.

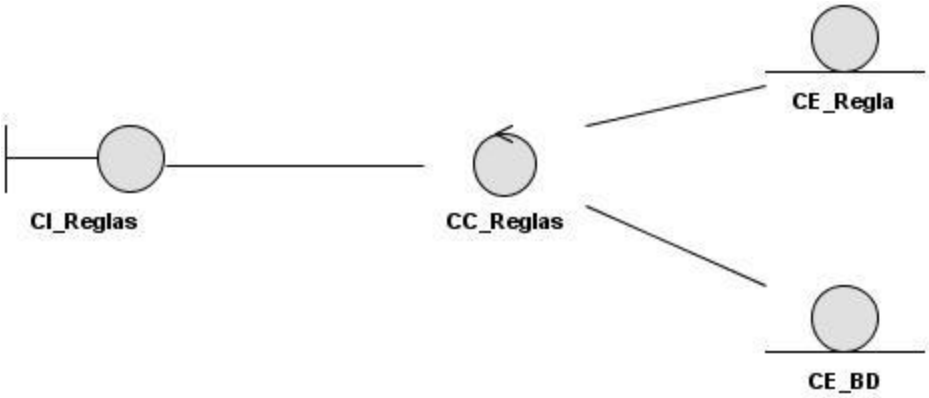


Fig. 11 Diagrama de Clases del Análisis CU Gestionar Reglas de los Destinos.

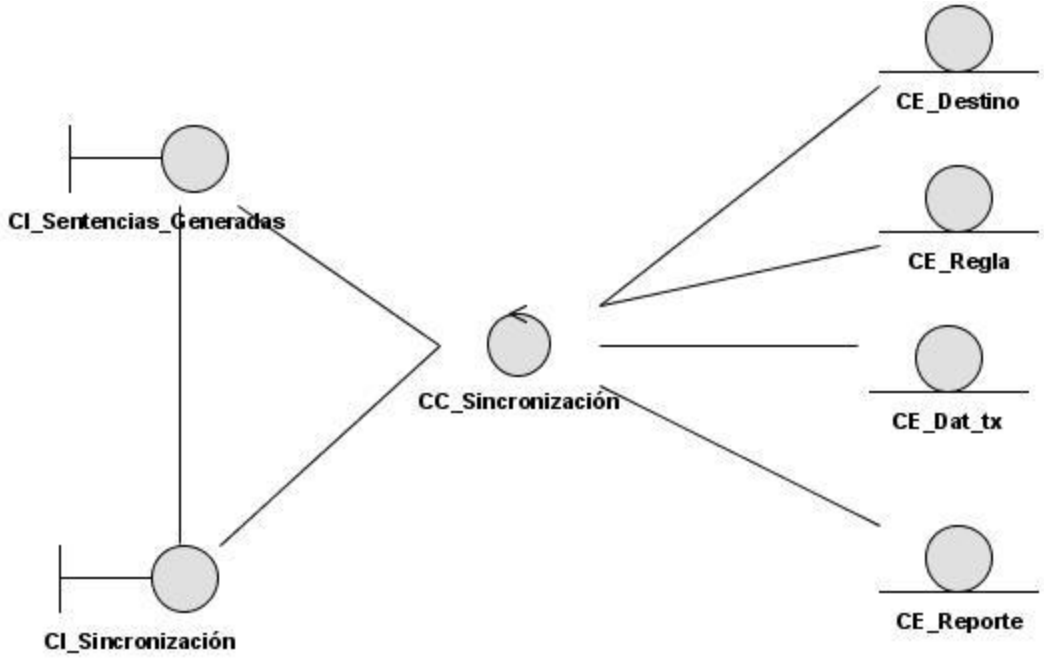


Fig. 12 Diagrama de Clases del Análisis CU Sincronizar.

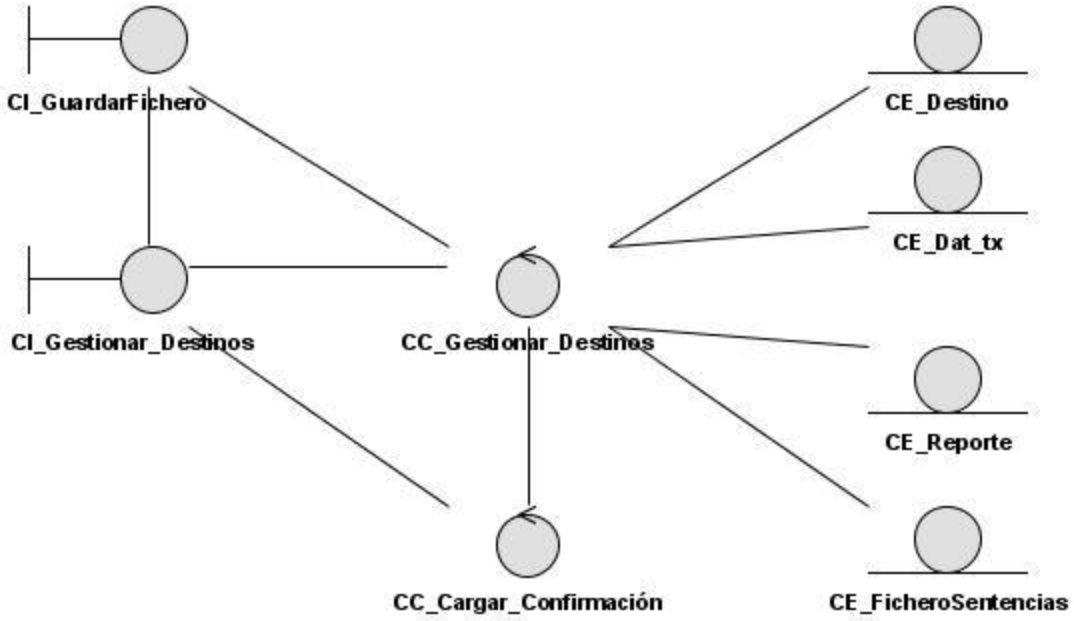


Fig. 13 Diagrama de Clases del Análisis CU Gestionar Destino de Datos.

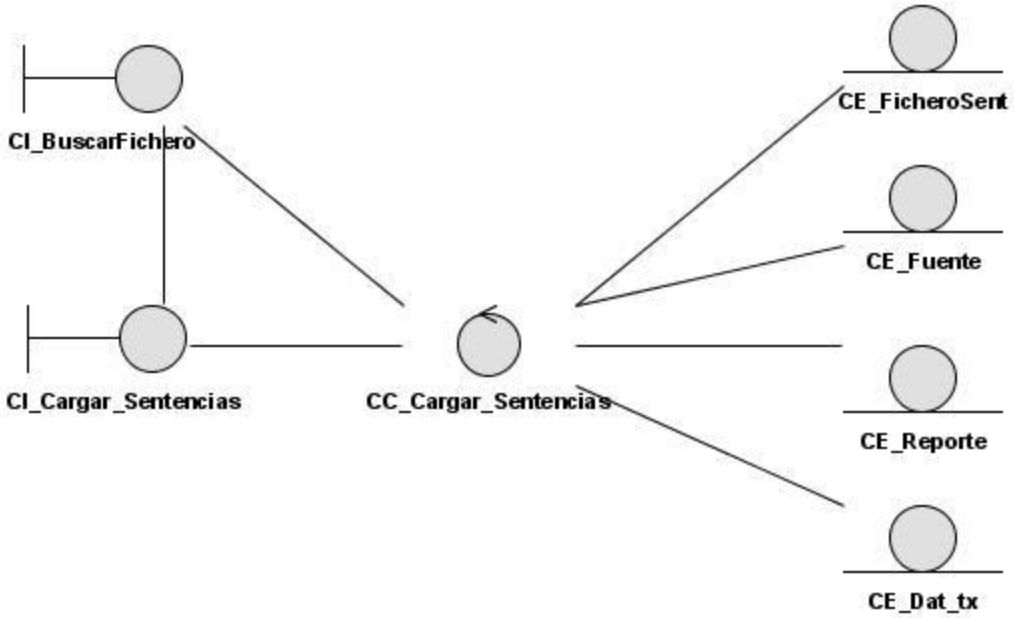


Fig. 14 Diagrama de Clases del Análisis CU Cargar Sentencias.

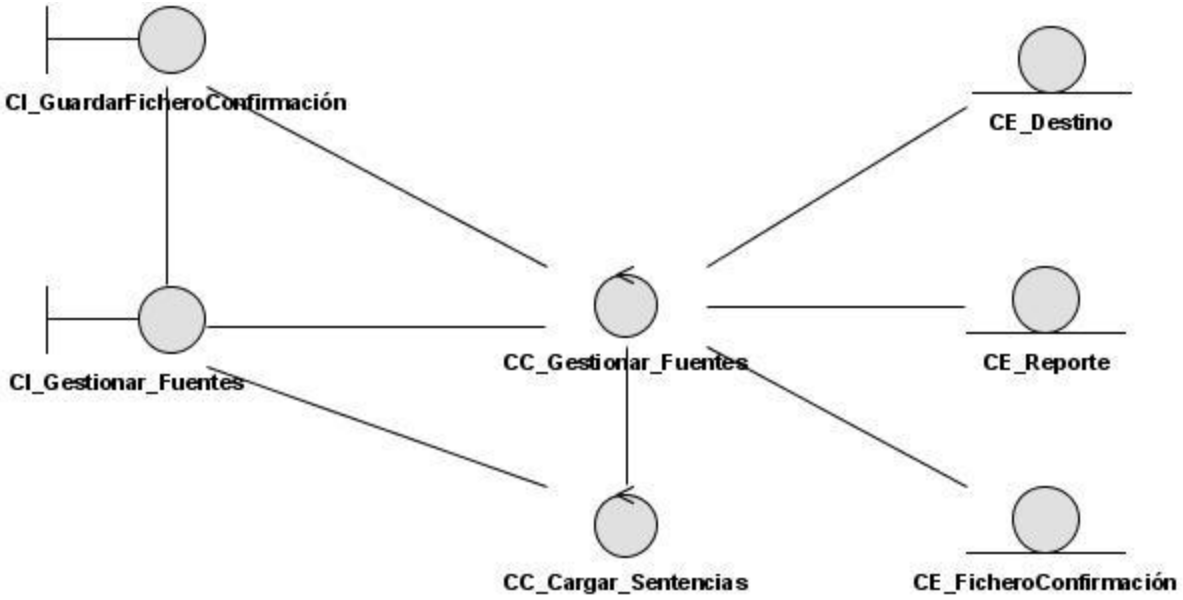


Fig. 15 Diagrama de Clases del Análisis CU Gestionar Fuente de Datos.

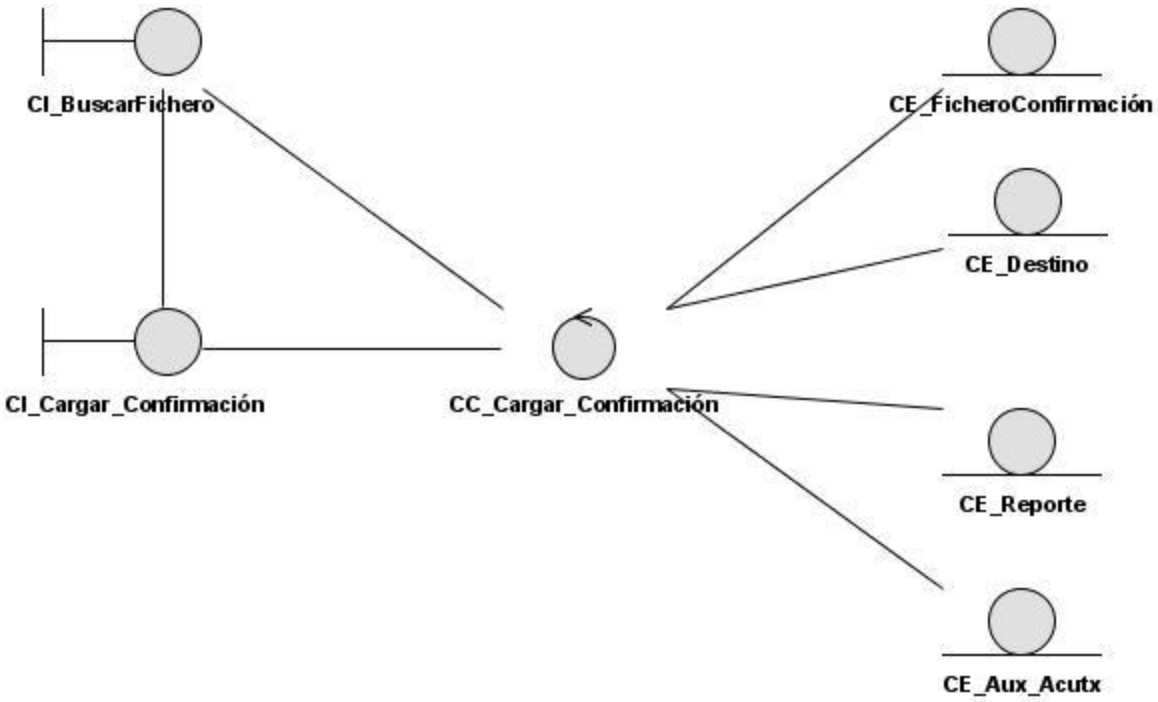


Fig. 16 Diagrama de Clases del Análisis CU Cargar Confirmación.

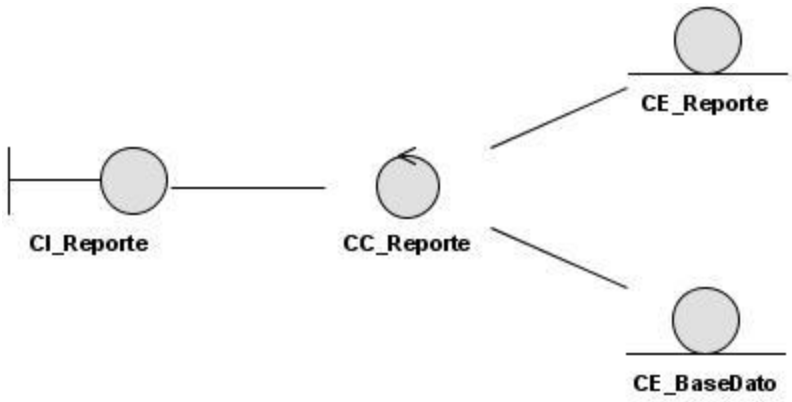


Fig. 17 Diagrama de Clases del Análisis CU Mostrar Reportes.

3.3 Diseño.

3.3.1 Descripción de la Arquitectura de la Aplicación.

La aplicación esta diseñada mediante la implementación de una arquitectura por capas con el objetivo de separar la lógica del diseño, la lógica del negocio y la lógica de datos.

La ventaja principal de este diseño es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de que sobrevenga algún cambio, sólo se ataca al nivel requerido sin tener que revisar entre un

código mezclado. Además, permite distribuir el trabajo de creación de una aplicación por niveles; de este modo, cada grupo de trabajo está totalmente abstraído del resto de niveles. [15]
En resumen se propone una aplicación que cuente con las siguientes capas:

1. Interfaz de Usuario.
2. Gestión de Interfaz (GI).
3. Negocio.
4. Acceso a datos (CAD).
5. Datos.

Capa de Interfaz de Usuario: Cuenta con un grupo de interfaces donde se capturan y se muestran los datos que los usuarios del sistema solicitan de cierta manera. Esta capa es netamente visual por lo que no ejecuta ningún código ni operaciones dentro de su estructura, solamente actúa como una capa de presentación. Cumple con los requisitos estándares de interfaz de usuario por lo que debe ser amigable, o sea: entendible y fácil de usar por el usuario.

Capa de Gestión de Interfaz: Une la capa de interfaz con la capa del negocio. Es la encargada de mapear los componentes de la capa de interfaz y capturar los eventos que se lancen en los componentes mapeados. Procesa estos eventos y realiza las principales operaciones relacionadas con el negocio del sistema. También se consideran aquí los programas de aplicación.

Capa de Negocio: Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Se comunica con la capa de datos a través de la capa de gestión de interfaz que procesa os datos capturados y mostrados al cliente. Usa la capa de datos a través de la capa de acceso a datos.

Capa de Acceso a Datos: Esta capa es usada por el modelo de negocio para la interacción con la base de datos, abstrayendo a esta de la forma que se realizan dichas operaciones. Es la encargada de solicitar al gestor de base de datos para almacenar o recuperar datos de él.

Capa de datos: Es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de acceso a datos.

3.3.2 Principios de Diseño a aplicar.

3.3.2.1 Pautas de Diseño de la Aplicación.

Para el diseño de las interfaces se usa un entorno de aplicación de escritorio donde el usuario interactúa con un grupo de ventanas que cuentan con un grupo de estándares de diseño como el color, la fuente y el tamaño de los componentes.

A continuación se muestra en la fig. 18 un ejemplo de prototipo de interfaz de usuario del sistema:

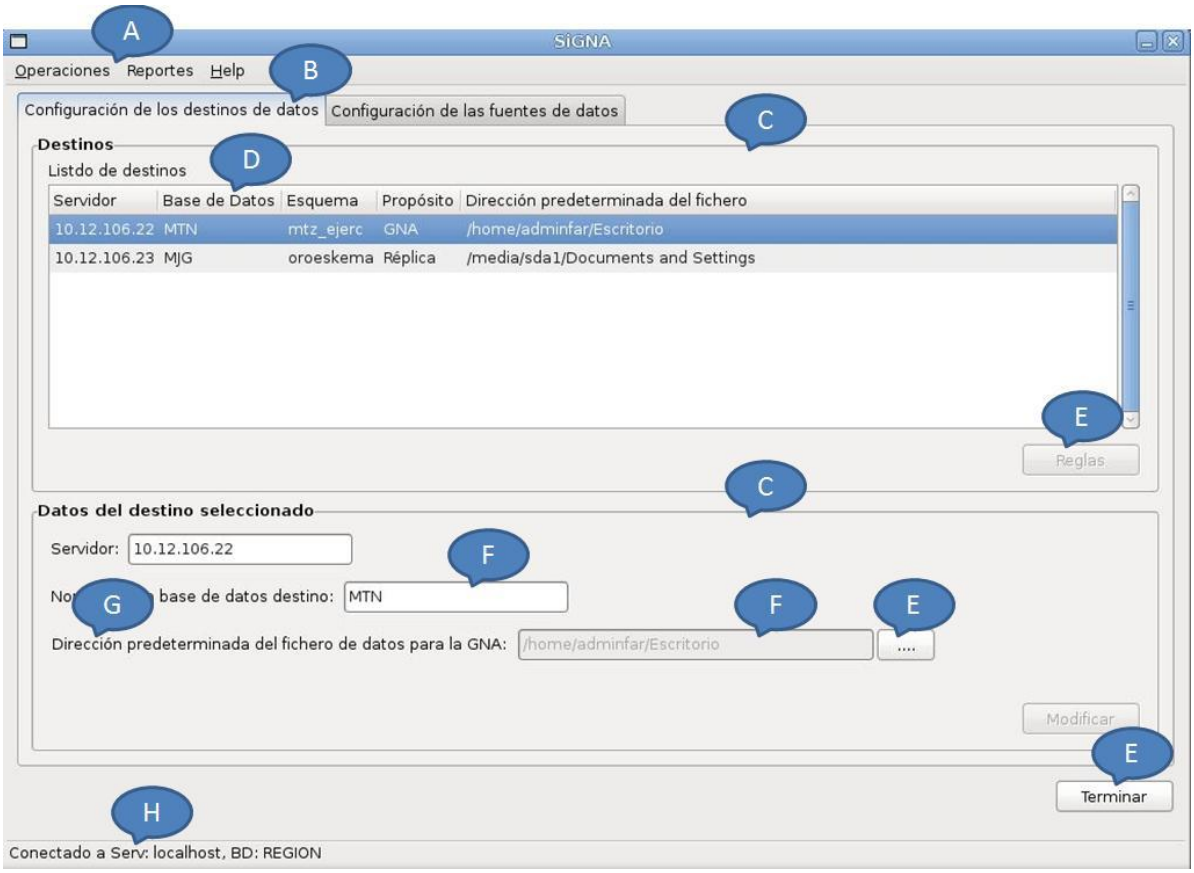


Fig. 18 Ejemplo de prototipo de interfaz de usuario del sistema.

Descripción de los componentes del prototipo de interfaz de Usuario:

Etiqueta	Objeto	Dimensiones	Tipo de Fuente	Tamaño Fuente
A	GTKMenuBar	-	Arial	12
B	GTKNoteBook	-	Arial	12
C	GTKFrame	-	Arial	12
D	GTKTreeView	-	Arial	12

E	GTKButton	100 X 28	Arial	12
F	GTKEntry	100 X 28	Arial	12
G	GTKLabel	100 X 28	Arial	12
H	GTKStatusBar	-	Arial	12

Tabla 12 Descripción de prototipos de interfaz de usuario.

Por otra parte los componentes con que cuenta el sistema en particular van a ser nombrados en diferencia a su tipo. Por ejemplo:

Botón: btn<nombre>.

Caja de Texto: txt<nombre>.

Etiqueta: lbl<nombre>.

Listado: lst<nombre>.

Para las instancias de objetos y archivos serán nombradas:

Formularios: frm<nombre>.

GI: gi<nombre>.

Negocio: n<nombre>.

CAD: cad<nombre>.

Para la declaración de las distintas clases:

Formularios: Frm<Nombre>.

Gestión de Interfaz: GI<Nombre>.

Negocio: N<Nombre>.

Capa de Acceso a datos: CAD<Nombre>.

3.3.3 Tratamiento de Errores.

Tratar los distintos errores es una acción considerada a lo largo del funcionamiento de la aplicación comenzando por datos entrados incorrectamente a mensajes de notificaciones por el mal uso de las acciones realizadas por el usuario.

Para lograr que el sistema que se desarrolla sea de fácil soporte y mantenimiento, debe adoptarse una estrategia apropiada para el tratamiento de excepciones. Al diseñarse un sistema, se debe garantizar que este sea capaz de:

- Detectar las excepciones.

- Mostrar una información detallada sobre la excepción detectada.

Cumpliéndose estas estrategias se garantiza un menor tiempo de comprensión y rectificación de los errores que puedan surgir que pueden ser internos del sistema o externos al entorno de trabajo.
El tratamiento de errores se realiza de forma uniforme a través de un componente que gestiona los errores y excepciones llamado “giError”, ubicado en la capa de Gestión de Interfaz (GI) del sistema. En él se capturan de forma general todos los eventos que han generado errores en tiempo de ejecución con sus respectivos mensajes y son mostrados en la interfaz, ver Fig. 19 y 20.
A continuación se muestra el prototipo de interfaz de usuario donde el sistema muestra los errores capturados:

Fig. 19 Ejemplo de pantalla de error.

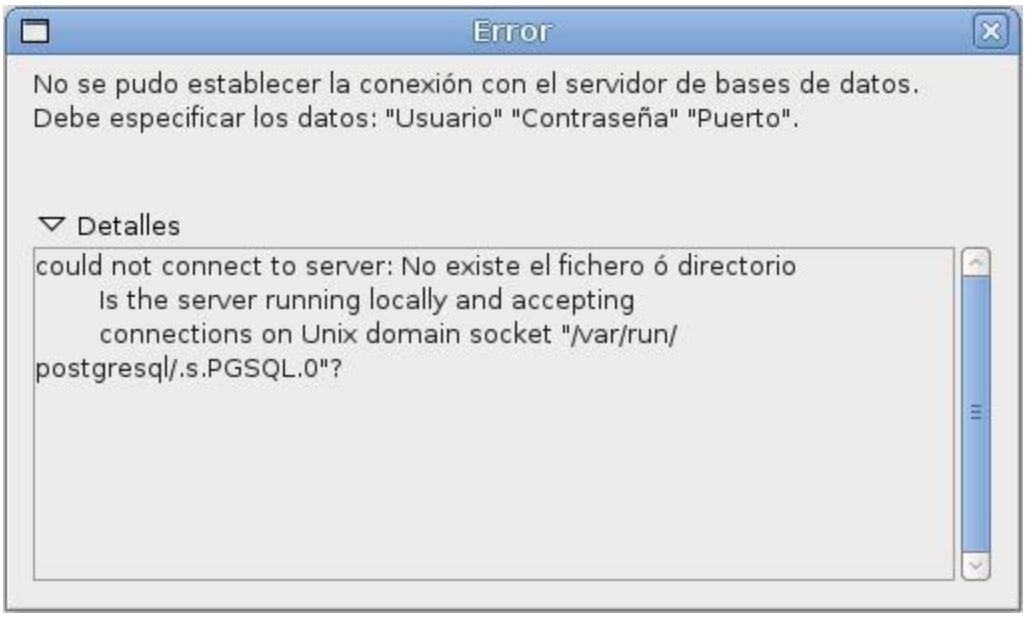


Fig. 20 Ejemplo de pantalla de error con detalles.

3.3.4 Diagrama de Clases del Diseño.

En este diagrama se representan las clases utilizadas dentro del sistema, las relaciones que existen entre ellas y los atributos y métodos pertenecientes a cada clase modelada. A continuación se muestran los diagramas de clases del diseño para cada caso de uso:

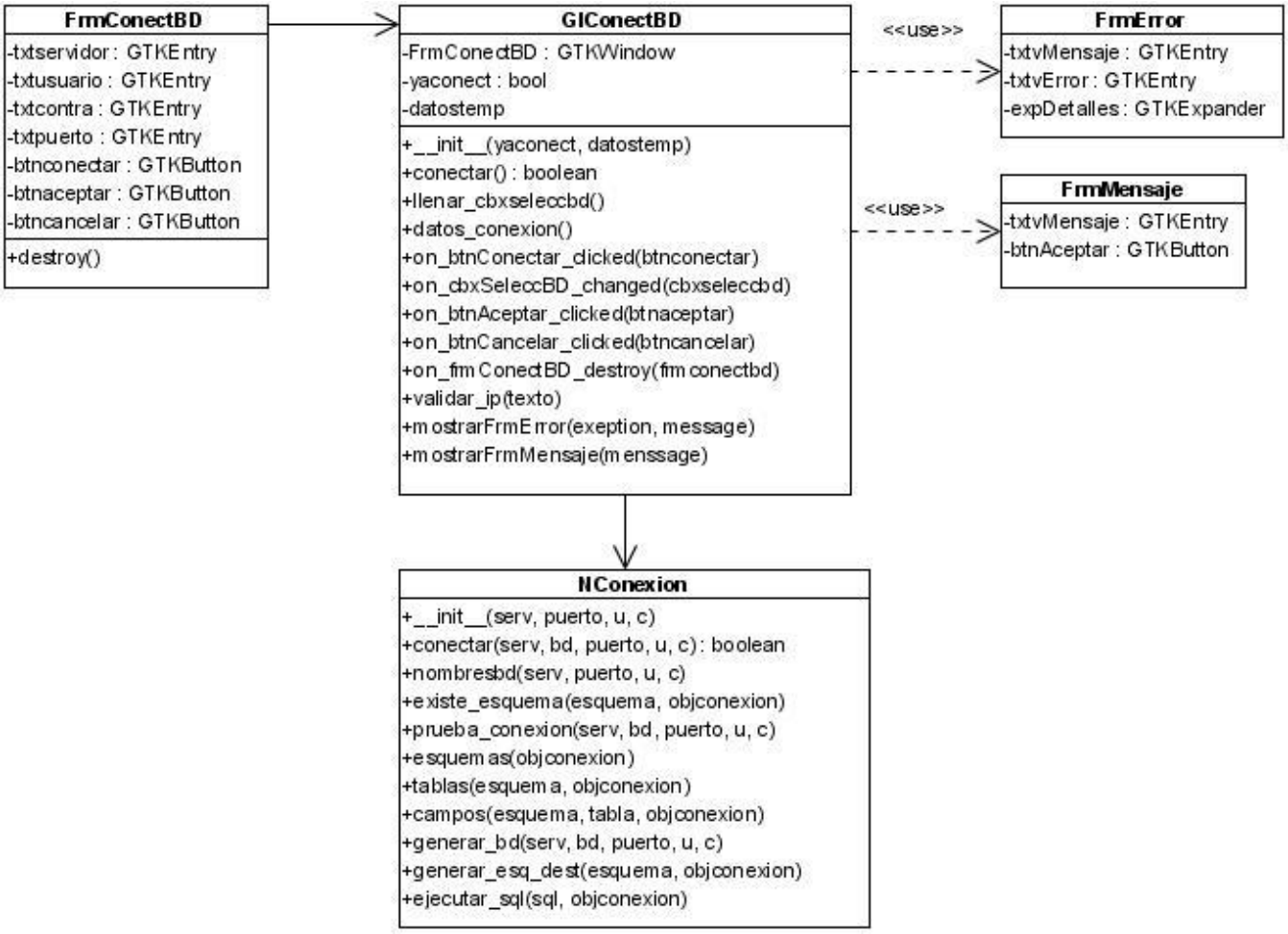


Fig. 21 Diagrama de Clases del Diseño CU Conectar BD.

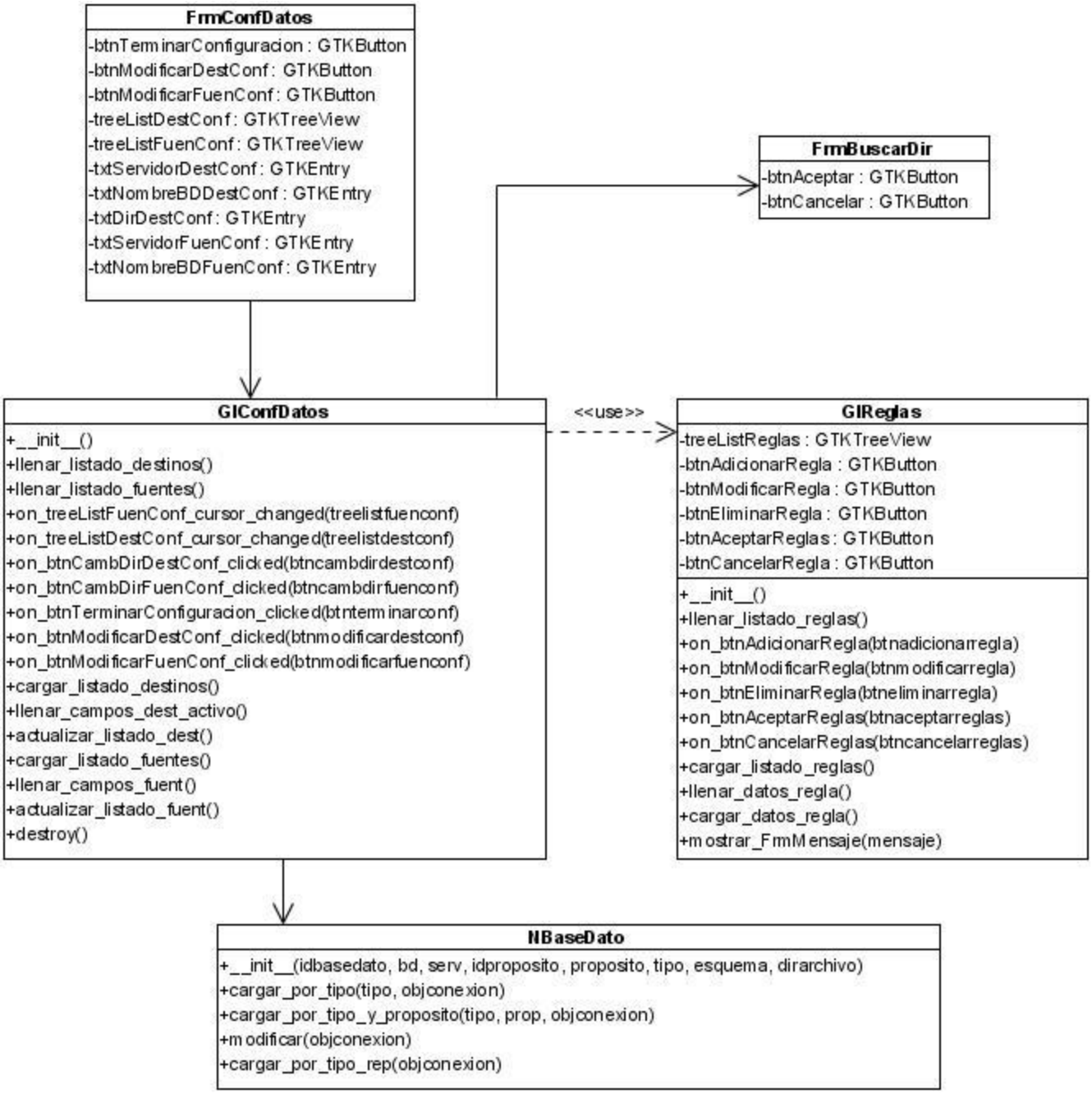


Fig. 22 Diagrama de Clases del Diseño CU Configurar Datos.

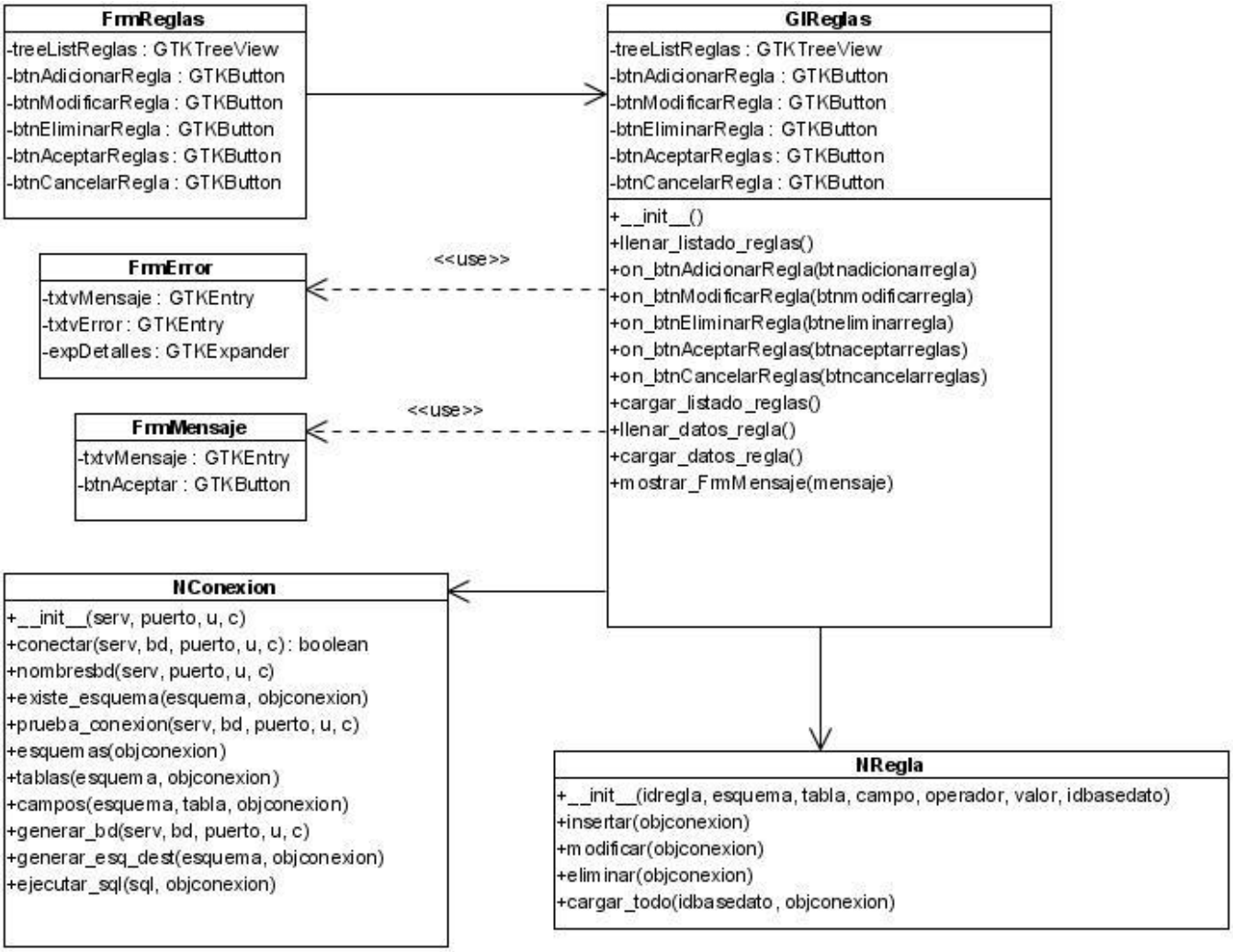


Fig. 23 Diagrama de Clases del Diseño CU Gestionar Reglas de los Destinos.

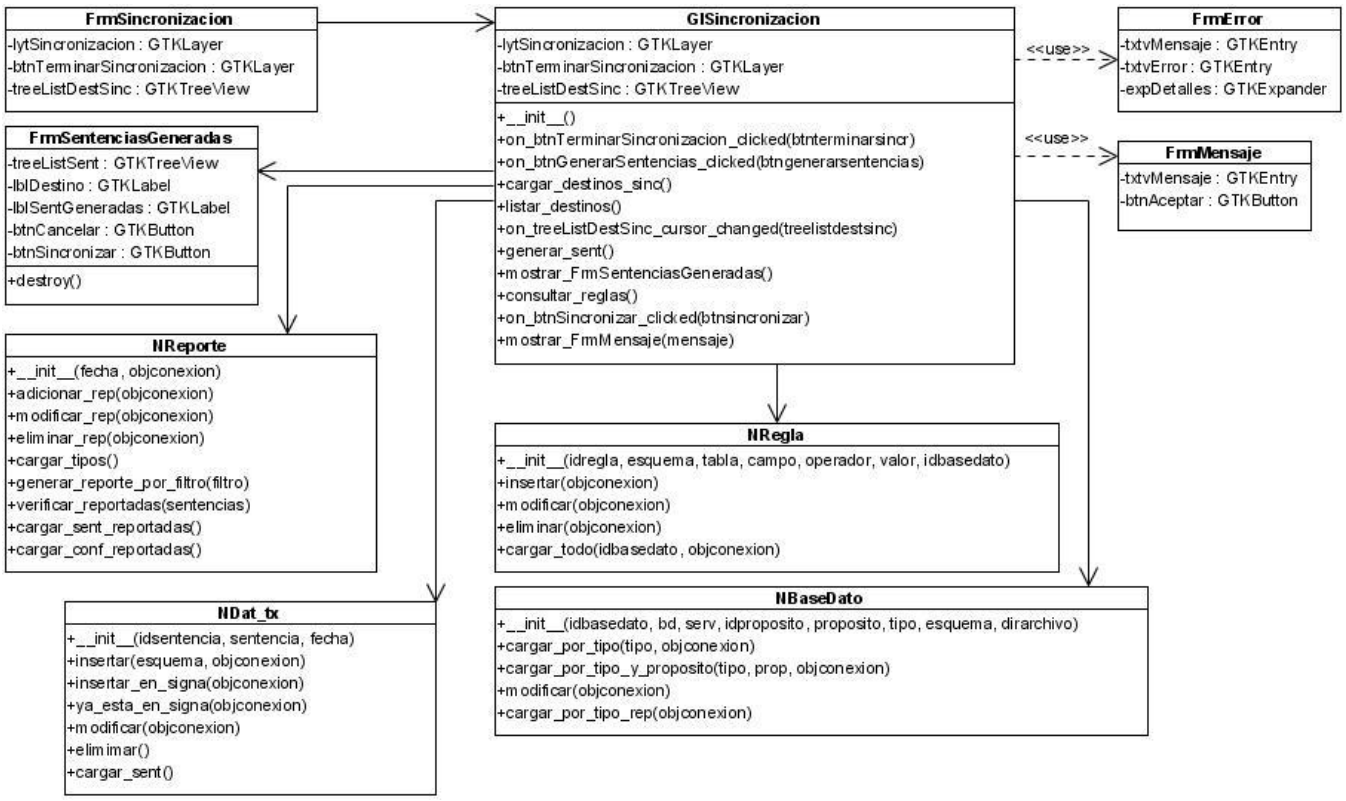


Fig. 24 Diagrama de Clases del Diseño CU Sincronizar.

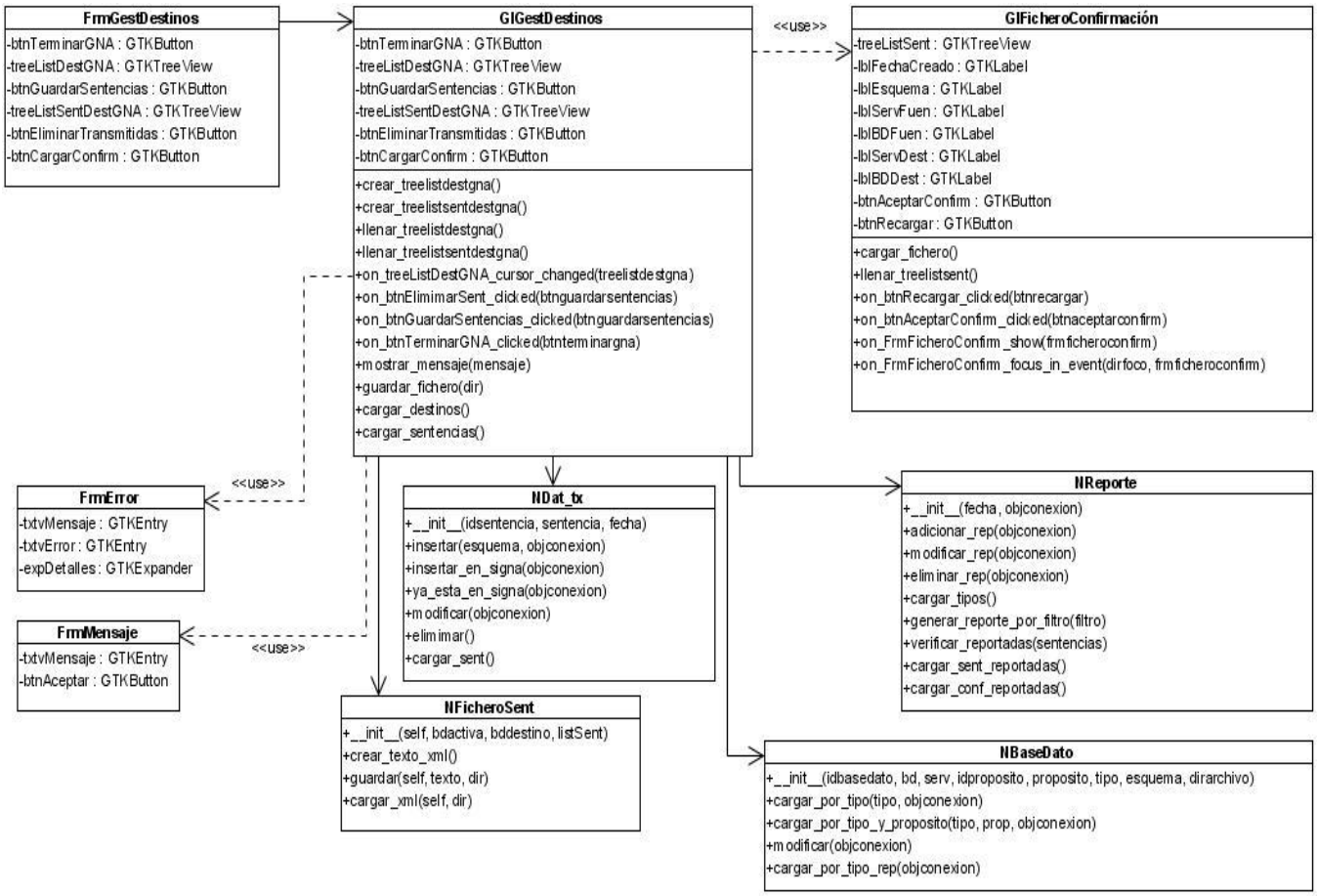


Fig. 25 Diagrama de Clases del Diseño CU Gestionar Destinos.

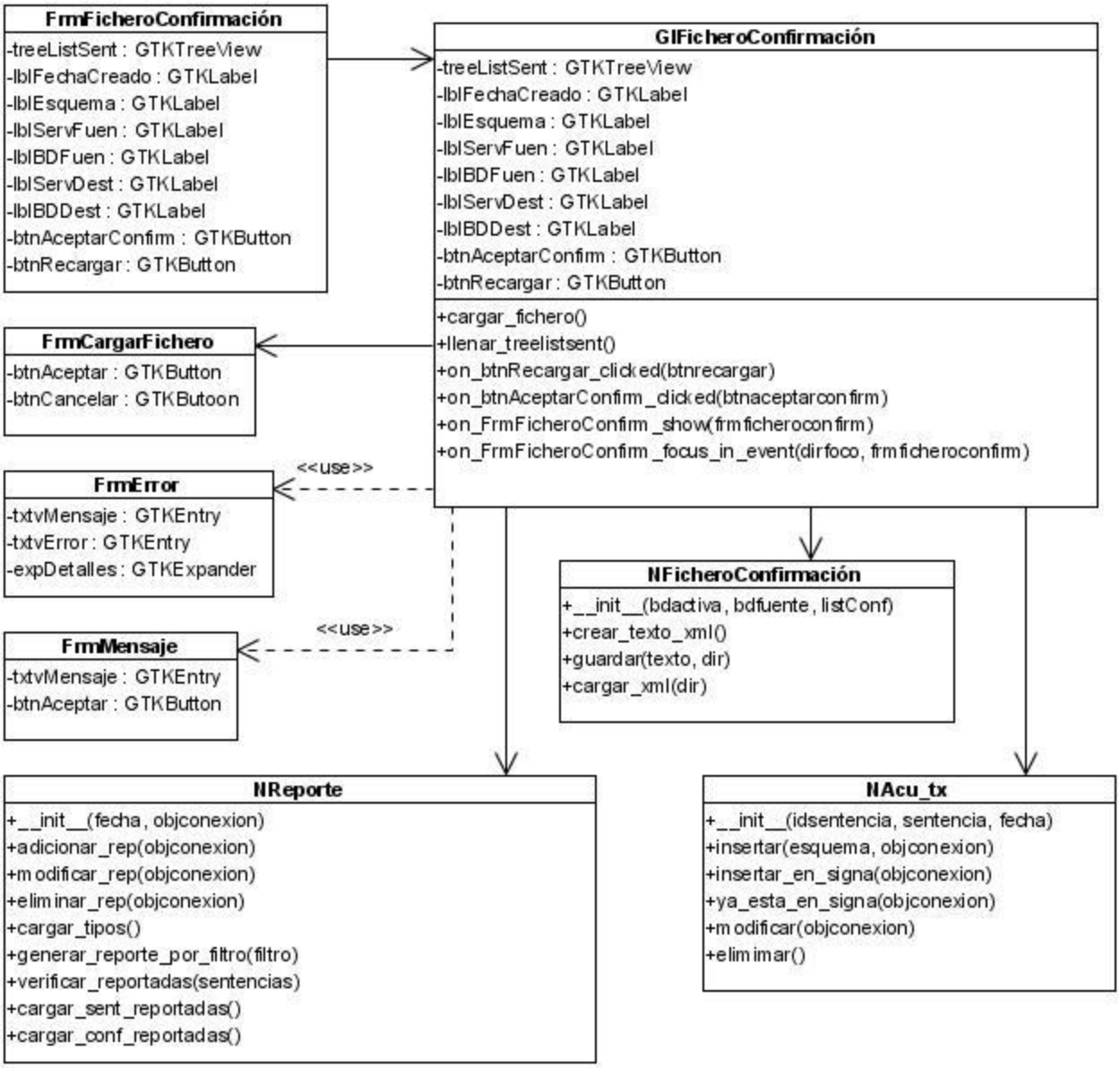


Fig. 26 Diagrama de Diseño CU Cargar Confirmación

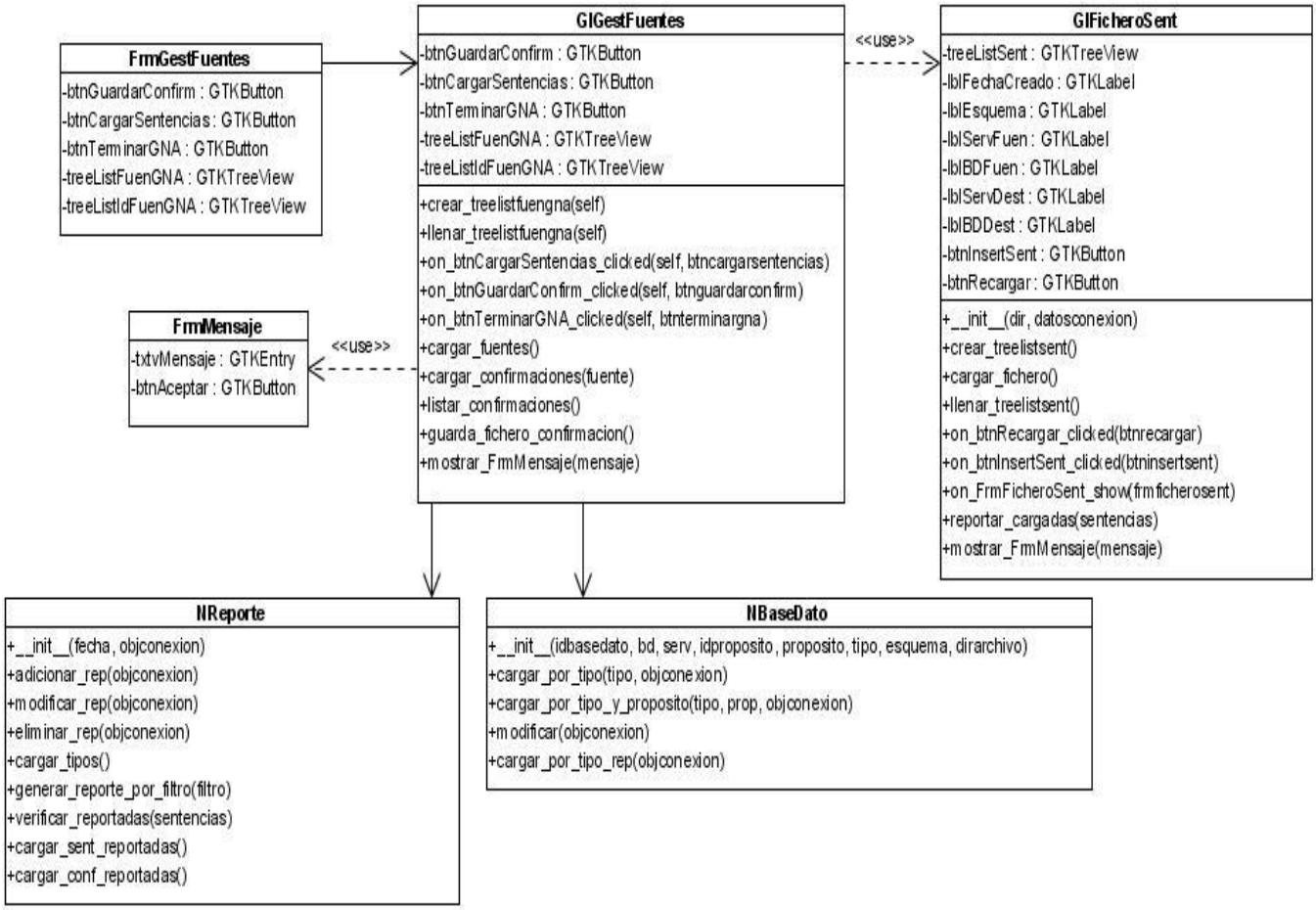


Fig. 27 Diagrama de Diseño CU Gestionar Fuentes

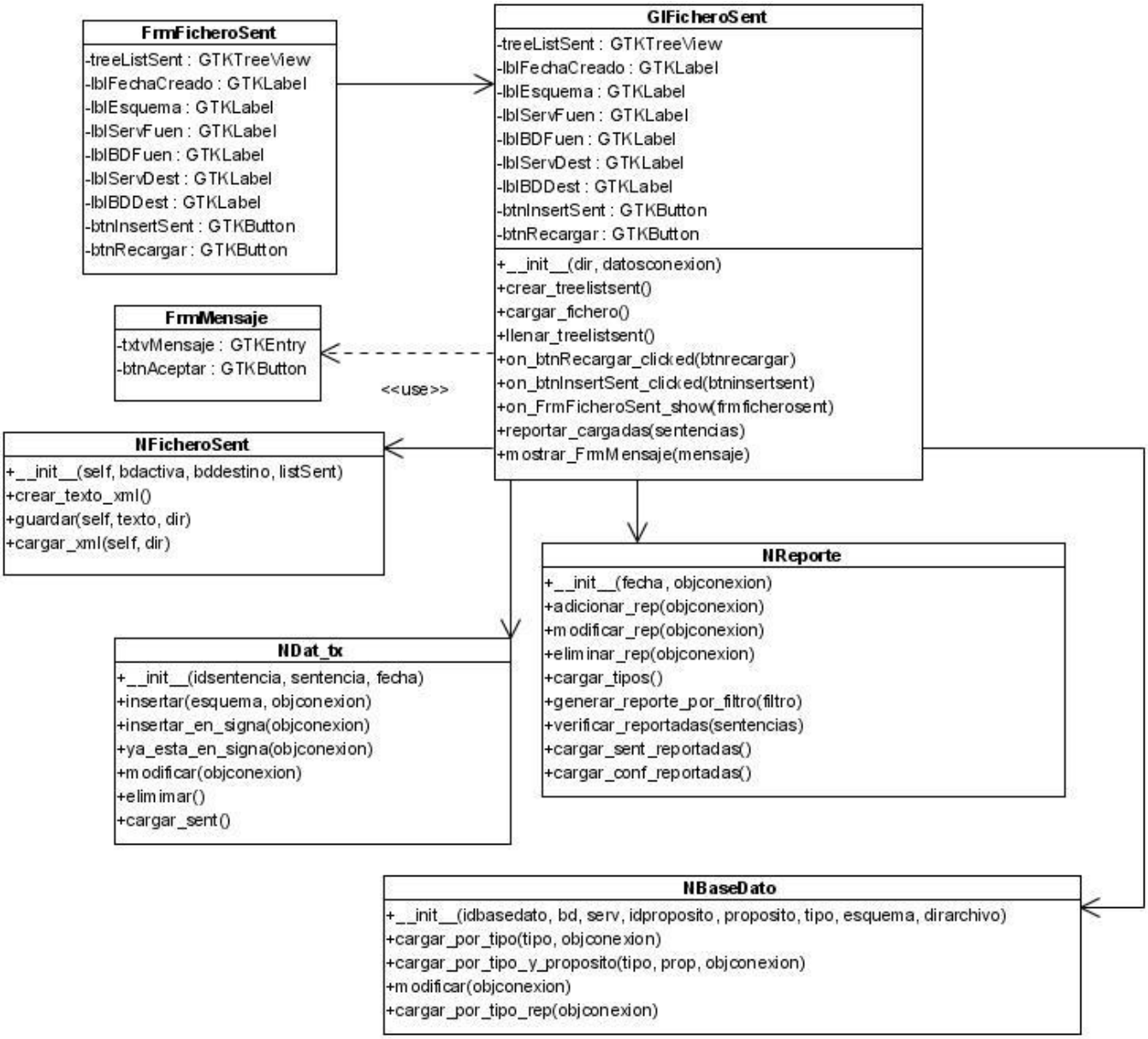


Fig. 28 Diagrama de Diseño CU Cargar Sentencias.

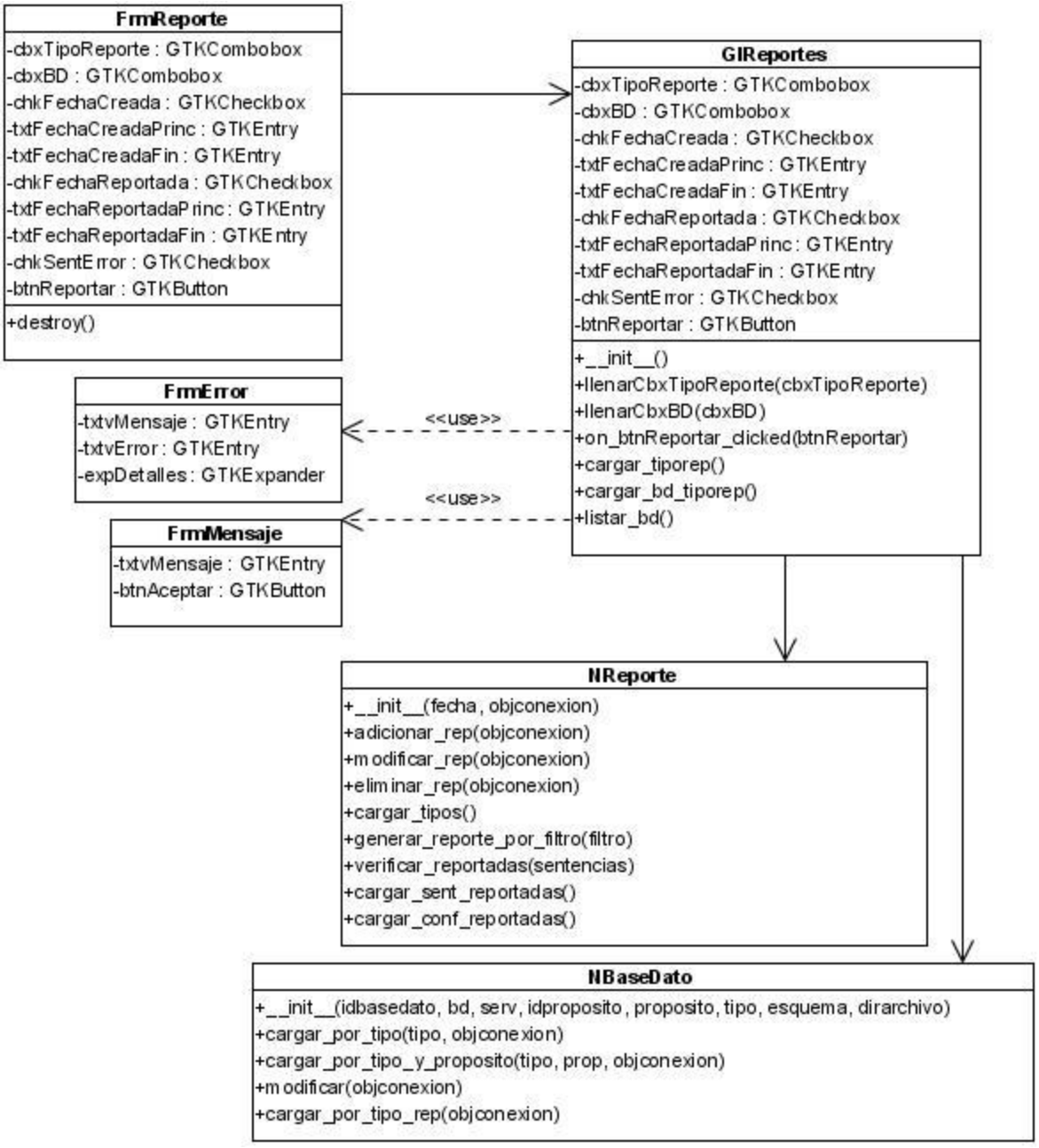


Fig. 29 Diagrama de Clases del Diseño CU Mostrar Reporte.

3.3.4.1 Descripción de Clases del Diseño.

Nombre: FrmConectBD	
Tipo de Clase: Interfaz	
Atributo:	**Tipo:**

txtservidor	GTKEntry
txtusuario	GTKEntry
txtcontra	GTKEntry
txtpuerto	GTKEntry
btnconectar	GTKButton
btnaceptar	GTKButton
btncancelar	GTKButton
Para cada Responsabilidad:	
Nombre:	destroy()
Descripción:	Destructor de la clase.

Tabla 13 Descripción de la Clase FrmConectBD.

Nombre: FrmError	
Tipo de Clase: Interfaz	
Atributo:	**Tipo:**
txtvMensaje	GTKEntry
txtvError	GTKEntry
expDetalles	GTKExpander
Para cada Responsabilidad:	
Nombre:	
Descripción:	

Tabla 14 Descripción de la Clase FrmError.

Nombre: FrmMensaje	
Tipo de Clase: Interfaz	
Atributo:	**Tipo:**
txtvMensaje	GTKEntry
btnAceptar	GTKButton
Para cada Responsabilidad:	
Nombre:	
Descripción:	

Tabla 15 Descripción de la Clase FrmMensaje

Nombre: FrmGestDestinos	
Tipo de Clase: Interfaz	
Atributo:	**Tipo:**
btnTerminarGNA	GTKButton
treeListDestGNA	GTKTreeView
treeListSentDestGNA	GTKTreeView
btnEliminarTransmitidas	GTKButton
btnCargarConfirm	GTKButton
Para cada Responsabilidad:	
Nombre:	
Descripción:	

Tabla 16 Descripción de la Clase FrmDestinos.

Nombre: GIConectBD	
Tipo de Clase: Gestión de Interfaz	
Atributo:	**Tipo:**
FrmConectBD	GTKWindow
yaconect	bool
datostemp	[]
Para cada Responsabilidad:	
Nombre:	__init__(yaconect, datostemp)
Descripción:	Constructor de la clase.
Nombre:	conectar()
Descripción:	Crea la conexiona a la base de datos.
Nombre:	llenar_cbxseleccbd()
Descripción:	Selecciona la base de datos en el combobox
Nombre:	datos_conexion()
Descripción:	Devuelve los datos de la conexión actual.
Nombre:	on_btnConectar_clicked(btnconectar)
Descripción:	Realiza la conexión a la base de datos seleccionada.
Nombre:	on_cbxSeleccBD_changed(cbxseleccbd)
Descripción:	Cambia la base de datos en el combobox.

Nombre:	on_btnAceptar_clicked(btnaceptar)
Descripción:	Acepta la conexión a la base de datos.
Nombre:	on_btnCancelar_clicked(btncancelar)
Descripción:	Cancela la conexión a la base de datos.
Nombre:	on_frmConectBD_destroy(frmconectbd)
Descripción:	Destruye la interfaz.
Nombre:	validar_ip(texto)
Descripción:	Valida la dirección IP del servidor.
Nombre:	mostrarFrmError(exeption, message)
Descripción:	Muestra la interfaz de Error.
Nombre:	mostrarFrmMensaje(menssage)
Descripción:	Muestra la interfaz del mensaje.

Tabla 17 Descripción de la Clase GIConectBD

Nombre: GIPrincipal	
Tipo de Clase: Gestión de Interfaz	
Atributo:	**Tipo:**
lytGNA	GTKLayer
btnTerminarGNA	GTKButton
treeListDestGNA	GTKTreeView
btnGuardarSentencias	GTKButton
btnCargarSentencias	GTKButton
treeListFuenGNA	GTKTreeView
treeListSentDestGNA	GTKTreeView
treeListIdFuenGNA	GTKTreeView
Para cada Responsabilidad:	
Nombre:	__init__(self)
Descripción:	Constructor de la clase.
Nombre:	mostrar_lytgna(self)
Descripción:	Muestra la interfaz de gestión de nodos aislados.
Nombre:	crear_treelistfuengna(self)
Descripción:	Crea el treeview de fuentes en la interfaz de la gestión de nodos aislados.

Nombre:	crear_treelistdestgna(self)
Descripción:	Crea el treeview de destinos en la interfaz de la gestión de nodos aislados.
Nombre:	crear_treelistsentdestgna(self)
Descripción:	Crea el treeview de sentencias en la interfaz de la gestión de nodos aislados.
Nombre:	llenar_treelistdestgna(self)
Descripción:	Llena el treeview de fuentes en la interfaz de la gestión de nodos aislados.
Nombre:	llenar_treelistfuengna(self)
Descripción:	Llena el treeview de destinos en la interfaz de la gestión de nodos aislados.
Nombre:	llenar_treelistsentdestgna(self)
Descripción:	Llena el treeview de sentencias en la interfaz de la gestión de nodos aislados.
Nombre:	on_treeListDestGNA_cursor_changed(self, treelistdestgna)
Descripción:	Muestra las sentencias correspondientes al destino seleccionado.
Nombre:	on_btnElimimarSent_clicked(self, btnguardarsentencias)
Descripción:	Elimina las sentencias seleccionadas.
Nombre:	on_btnCargarSentencias_clicked(self, btnguardarsentencias)
Descripción:	Muestra la interfaz de cargar las sentencias.
Nombre:	on_btnGuardarSentencias_clicked(self, btnguardarsentencias)
Descripción:	Muestra el interfaz de guardar las sentencias.
Nombre:	on_btnTerminarGNA_clicked(self, btnterminargna)
Descripción:	Termina el proceso de gestión de nodos aislados y cierra la interfaz.
Nombre:	on_menGNA_activate(self, mengna)
Descripción:	Muestra el mensaje de confirmación de gestión de nodos aislados.

Tabla 18 Descripción de la Clase GIPrincipal.

Nombre: GIConfDatos	
Tipo de Clase: Gestión de Interfaz	
Atributo:	**Tipo:**
btnTerminarConfiguracion	GTKButton
btnModificarDestConf	GTKButton
btnModificarFuenConf	GTKButton

treeListDestConf	GTKTreeView
treeListFuenConf	GTKTreeView
txtServidorDestConf	GTKEntry
txtNombreBDDestConf	GTKEntry
txtDirDestConf	GTKEntry
txtServidorFuenConf	GTKEntry
txtNombreBDFuenConf	GTKEntry
Para cada Responsabilidad:	
Nombre:	__init__(self)
Descripción:	Constructor de la clase.
Nombre:	mostrar_lytconfiguracion(self)
Descripción:	Muestra la pagina de configuración de datos.
Nombre:	crear_treelistdestconf(self)
Descripción:	Crea el treeview de configuración de los destinos.
Nombre:	crear_treelistfuenconf(self)
Descripción:	Crea el treeview de configuración de las fuentes.
Nombre:	llenar_treelistdestconf(self)
Descripción:	Lista en el treeview los destinos existentes en la configuración.
Nombre:	llenar_treelistfuenconf(self)
Descripción:	Lista en el treeview las fuentes existentes en la configuración.
Nombre:	on_txtServidorFuenConf_changed(self, txtservidorfuenconf)
Descripción:	Cambia el nombre del servidor en la fuente seleccionada.
Nombre:	on_txtNombreBDFuenConf_changed(self, txtnombrebdfuenconf)
Descripción:	Cambia el nombre de la base de datos de la fuente seleccionada.
Nombre:	on_txtServidorDestConf_changed(self, txtservidordestconf)
Descripción:	Cambia el nombre del servidor en el destino seleccionado.
Nombre:	on_txtNombreBDDestConf_changed(self, txtnombrebddestconf)
Descripción:	Cambia el nombre de la base de datos del destino seleccionado.
Nombre:	on_treeListFuenConf_cursor_changed(self, treelistfuenconf)
Descripción:	Muestra los datos de la fuente seleccionada por el cursor.
Nombre:	on_treeListDestConf_cursor_changed(self, treelistdestconf)
Descripción:	Muestra los datos del destino seleccionado por el cursor.

Nombre:	on_btnCambDirDestConf_clicked(self, btncambdirdestconf)
Descripción:	Cambia la dirección donde se guarda el fichero con los las sentencias.
Nombre:	on_btnCambDirFuenConf_clicked(self, btncambdirfuenconf)
Descripción:	Cambia la dirección donde se cargará el fichero con los las sentencias.
Nombre:	on_btnTerminarConfiguracion_clicked(self, btnterminarconf)
Descripción:	Termina el proceso de configuración de datos.
Nombre:	on_btnModificarDestConf_clicked(self, btnmodificardestconf)
Descripción:	Modifica los datos del destino seleccionado y actualiza el treeview de destinos.
Nombre:	on_btnModificarFuenConf_clicked(self, btnmodificarfuenconf)
Descripción:	Modifica los datos de la fuente seleccionada y actualiza el treeview de fuentes.
Nombre:	on_menConfiguracion_activate(self, menconfiguracion)
Descripción:	Muestra el mensaje de confirmación de los cambios en la configuración de datos.

Tabla 19 Descripción de la Clase GIConfDatos.

Nombre: GISincronizacion	
Tipo de Clase: Gestión de Interfaz	
Atributo:	**Tipo:**
lytSincronizacion	GTKLayer
btnTerminarSincronizacion	GTKLayer
treeListDestSinc	GTKTreeView
Para cada Responsabilidad:	
Nombre:	__init__(self, dir, datosconexion)
Descripción:	Constructor de la clase.
Nombre:	mostrar_lytsincronizacion(self)
Descripción:	Muestra el layer de sincronización.
Nombre:	on_btnTerminarSincronizacion_clicked(self, btnterminarsincr)
Descripción:	Termina el proceso de sincronización.
Nombre:	on_menSincInic_activate(self, mensincinic)
Descripción:	Activa el mensaje de sincronización inicial realizada.

Nombre:	on_btnGenerarSentencias_clicked(self, btngenerarsentencias)
Descripción:	Genera las sentencias para la sincronización

Tabla 20 Descripción de la Clase GISincronizacion.

Nombre: GISentencias	
Tipo de Clase: Gestión de Interfaz	
Atributo:	**Tipo:**
treeListSent	GTKTreeView
lblDestino	GTKLabel
lblTotalSent	GTKLabel
btnSincronizar	GTKButton
btnCancelar	GTKButton
Para cada Responsabilidad:	
Nombre:	__init__(self, dir, datosconexion)
Descripción:	Constructor de la clase.
Nombre:	crear_treelistsent(self)
Descripción:	Crea el treeview del listado de sentencias.
Nombre:	llenar_treelistsent(self)
Descripción:	Llena el treeview de sentencias.
Nombre:	generar_sent(self)
Descripción:	Genera las sentencias.
Nombre:	indices_reglas_asociadas(self, regla, lista)
Descripción:	Muestra la lista de reglas asociadas.
Nombre:	on_btnCancelar_clicked(self, btncancelar)
Descripción:	Cancela el proceso de sincronizar.
Nombre:	on_btnSincronizar_clicked(self, btnsincronizar)
Descripción:	Realiza el proceso de sincronizar.

Tabla 21 Descripción de la Clase GISentencias.

Nombre: GIFicheroSent	
Tipo de Clase: Gestión de Interfaz	
Atributo:	**Tipo:**

treeListSent	GTKTreeView
lblFechaCreado	GTKLabel
lblEsquema	GTKLabel
lblServFuen	GTKLabel
lblBDFuen	GTKLabel
lblServDest	GTKLabel
lblBDDest	GTKLabel
btnInsertSent	GTKButton
btnRecargar	GTKButton
Para cada Responsabilidad:	
Nombre:	__init__(self, dir, datosconexion)
Descripción:	Constructor de la clase.
Nombre:	crear_treelistsent(self)
Descripción:	Crea el treeview sentencias.
Nombre:	cargar_fichero(self)
Descripción:	Carga el fichero de sentencias.
Nombre:	llenar_treelistsent(self)
Descripción:	Llena el treeview con las sentencias cargadas.
Nombre:	on_btnRecargar_clicked(self, btnrecargar)
Descripción:	Recarga las sentencias del fichero.
Nombre:	on_btnInsertSent_clicked(self, btninsertsent)
Descripción:	Inserta las sentencias en la BD.
Nombre:	on_FrmFicheroSent_show(self, frmficherosent)
Descripción:	Muestra el formulario fichero sentencias.

Tabla 22 Descripción de la Clase GIFicheroSent.

Nombre: GIReglas	
Tipo de Clase: Gestión de Interfaz	
Atributo:	**Tipo:**
treeListReglas	GTKTreeView
btnAdicionarRegla	GTKButton
btnModificarRegla	GTKButton

btnEliminarRegla	GTKButton
btnAceptarReglas	GTKButton
btnCancelarRegla	GTKButton
Para cada Responsabilidad:	
Nombre:	__init__(self, mensaje, cerrar)
Descripción:	Constructor de la clase.
Nombre:	crear_treeListReglas(self)
Descripción:	Crea el TreeView de reglas.
Nombre:	llenar_treeListReglas(self)
Descripción:	Llena el TreeView de reglas.
Nombre:	on_btnAdicionarRegla(self, btnadicionarregla)
Descripción:	Adiciona una regla al listado.
Nombre:	on_btnModificarRegla(self, btnmodificarregla)
Descripción:	Modifica la regla seleccionada del listado..
Nombre:	on_btnEliminarRegla(self, btneliminarregla)
Descripción:	Elimina la regla seleccionada del listado.
Nombre:	on_btnAceptarReglas(self, btnaceptarreglas)
Descripción:	Acepta la operación de gestionar reglas.
Nombre:	on_btnCancelarReglas(self, btncancelarreglas)
Descripción:	Cancela la operación de gestionar reglas.

Tabla 23 Descripción de la Clase GIReglas.

Nombre: GIMensaje.	
Tipo de Clase: Gestión de Interfaz	
Atributo:	**Tipo:**
txtvMensaje	GTKEntry
btnAceptar	GTKButton
Para cada Responsabilidad:	
Nombre:	__init__(self, mensaje, cerrar)
Descripción:	Constructor de la clase.
Nombre:	on_btnAceptar_clicked(self, btnaceptar)
Descripción:	Acepta y oculta el interfaz de mensaje.

Nombre:	on_FrmMensaje_destroy(self, frmm)
Descripción:	Destruye el interfaz de mensaje.

Tabla 24 Descripción de la Clase GIMensaje.

Nombre: GIError	
Tipo de Clase: Gestión de Interfaz	
Atributo:	**Tipo:**
txtvMensaje	GTKEntry
txtvError	GTKEntry
expDetalles	GTKExpander
Para cada Responsabilidad:	
Nombre:	__init__(self, mensage, error)
Descripción:	Constructor de la clase.
Nombre:	expanded_callback(self, expdetalles, param_spec)
Descripción:	Muestra y oculta los detalles del error.

Tabla 25 Descripción de la Clase GIError.

Nombre: GIMensajeCondicional	
Tipo de Clase: Gestión de Interfaz	
Atributo:	**Tipo:**
txtvMensaje	GTKEntry
btnAceptar	GTKButton
btnCancelar	GTKButton
Para cada Responsabilidad:	
Nombre:	__init__(self, mensage)
Descripción:	Constructor de la clase.
Nombre:	on_btnAceptar_clicked(self, btnaceptar)
Descripción:	Muestra y oculta los detalles del error.
Nombre:	on_btnCancelar_clicked(self, btncancelar)
Descripción:	Cierra la interfaz del mensaje y no realiza ningún cambio.
Nombre:	on_FrmMensajeCondicional_destroy(self, frmmcond)
Descripción:	Destruye la interfaz del mensaje.

Tabla 26 Descripción de la Clase GIMensajeCondicional.

Nombre: GIBuscarDir.	
Tipo de Clase: Gestión de Interfaz	
Atributo:	**Tipo:**
btnAceptar	GTKButton
btnCancelar	GTKButton
Para cada Responsabilidad:	
Nombre:	__init__(self)
Descripción:	Constructor de la clase.
Nombre:	on_btnAceptar_clicked(self, btnaceptar)
Descripción:	Toma la dirección seleccionada y oculta la interfaz.

Tabla 27 Descripción de la Clase GIBuscarDir.

Nombre: GIReporte	
Tipo de Clase: Gestión de Interfaz	
Atributo:	**Tipo:**
-cbxTipoReporte	GTKCombobox
-cbxBD	GTKCombobox
-chkFechaCreada	GTKCheckbox
-txtFechaCreadaPrinc	GTKEntry
-txtFechaCreadaFin	GTKEntry
-chkFechaReportada	GTKCheckbox
-txtFechaReportadaPrinc	GTKEntry
-txtFechaReportadaFin	GTKEntry
-chkSentError	GTKCheckbox
-btnReportar	GTKButton
Para cada Responsabilidad:	
Nombre:	__init__(self, mensaje, cerrar)
Descripción:	Constructor de la clase.
Nombre:	llenarCbxTipoReporte(self, cbxTipoReporte)
Descripción:	Llena el combobox con los tipos de reportes.
Nombre:	llenarCbxBD(self, tiporeporte, cbxBD)
Descripción:	Llena el combobox con las BD existentes para ese tipo de reporte

Nombre:	on_btnReportar_clicked(self, btnReportar)
Descripción:	Muestra el reporte seleccionado.

Tabla 28 Descripción de la Clase GIReporte.

Nombre: NConexion	
Tipo de Clase: Negocio	
Atributo:	**Tipo:**
Para cada Responsabilidad:	
Nombre:	__init__(serv, puerto, u, c)
Descripción:	Constructor de la clase.
Nombre:	conectar(serv, bd, puerto, u, c)
Descripción:	Crea la conexión a la base de datos.
Nombre:	nombresbd(serv, puerto, u, c)
Descripción:	Lista las bases de datos existentes en el servidor correspondiente.
Nombre:	existe_esquema(esquema, objconexion)
Descripción:	Comprueba si el esquema “Signa existe en la BD.”
Nombre:	prueba_conexion(serv, bd, puerto, u, c)
Descripción:	Prueba la conexión con el servidor de datos.
Nombre:	esquemas(objconexion)
Descripción:	Devuelve un listado de esquemas.
Nombre:	tablas(esquema, objconexion)
Descripción:	Devuelve las tablas de una BD.
Nombre:	campos(esquema, tabla, objconexion)
Descripción:	Devuelve los campos de una BD.
Nombre:	generar_bd(serv, bd, puerto, u, c)
Descripción:	Genera el esquema necesario para el funcionamiento del software.
Nombre:	generar_esq_dest(esquema, objconexion)
Descripción:	Genera un esquema para la replicación en caso de que se agregue un nuevo destino.
Nombre:	ejecutar_sql(sql, objconexion)
Descripción:	Ejecuta las consultas SQL pasadas por parámetros.

Tabla 29 Descripción de la Clase NConexion.

Nombre: NFicheroSent	
Tipo de Clase: Negocio	
Atributo:	**Tipo:**
Para cada Responsabilidad:	
Nombre:	__init__(self, bdactiva, bddestino, listSent)
Descripción:	Constructor de la clase.
Nombre:	crear_texto_xml(self)
Descripción:	Crea el texto en formato XML para insertar en el fichero.
Nombre:	guardar(self, texto, dir)
Descripción:	Guarda el texto con los datos de las sentencias dentro del fichero XML.
Nombre:	cargar_xml(self, dir)
Descripción:	Carga el fichero XML y guarda los datos en el sistema.

Tabla 30 Descripción de la Clase NFicheroSent.

Nombre: NBaseDato	
Tipo de Clase: Negocio	
Atributo:	**Tipo:**
Para cada Responsabilidad:	
Nombre:	__init__(self, idbasedato, bd, serv, idproposito, proposito, tipo, esquema, dirarchivo)
Descripción:	Constructor de la clase.
Nombre:	cargar_por_tipo(self, tipo, objconexion)
Descripción:	Carga las fuentes y los destinos por por el tipo de acción.
Nombre:	cargar_por_tipo_y_proposito(self, tipo, prop, objconexion)
Descripción:	Carga las fuentes y los destinos por el tipo de propósito.
Nombre:	modificar(objconexion)
Descripción:	Modifica una tupla en la tabla correspondiente.
Nombre:	cargar_por_tipo_rep(objconexion)
Descripción:	Carga los reportes por un tipo específico.

Tabla 31 Descripción de la Clase NBaseDato.

Nombre: NDat_tx	
Tipo de Clase: Negocio	
Atributo:	**Tipo:**
Para cada Responsabilidad:	
Nombre:	__init__(self, idsentencia, sentencia, fecha)
Descripción:	Constructor de la clase.
Nombre:	insertar(self, esquema, objconexion)
Descripción:	Insertar en la tabla Dat_tx.
Nombre:	insertar_en_signa(self, objconexion)
Descripción:	Instertar en las tablas del esquema “SiGNA”.
Nombre:	ya_esta_en_signa(self, objconexion)
Descripción:	Verifica si la tabla esta en el esquema “SiGNA”.
Nombre:	modificar(self, objconexion)
Descripción:	Modifica en la tabla Dat_tx.
Nombre:	elimimar(self)
Descripción:	Elimina en la tabla Dat_tx.
Nombre:	cargar_sent()
Descripción:	Carga las sentencias que se encuentran dentro de la tabla de transmisión.

Tabla 32 Descripción de la Clase NDat_tx.

Nombre: NAcu_tx	
Tipo de Clase: Negocio	
Atributo:	**Tipo:**
Para cada Responsabilidad:	
Nombre:	__init__(self, idsentencia, sentencia, fecha)
Descripción:	Constructor de la clase.
Nombre:	insertar(self, esquema, objconexion)
Descripción:	Insertar en la tabla Dat_tx.
Nombre:	insertar_en_signa(self, objconexion)
Descripción:	Instertar en las tablas del esquema “SiGNA”.

Nombre:	ya_esta_en_signa(self, objconexion)
Descripción:	Verifica si la tabla esta en el esquema “SiGNA”.
Nombre:	modificar(self, objconexion)
Descripción:	Modifica en la tabla Dat_tx.
Nombre:	elimimar(self)
Descripción:	Elimina en la tabla Dat_tx.

Tabla 33 Descripción de la Clase NAcu_tx.

Nombre: NRegla	
Tipo de Clase: Negocio	
Atributo:	**Tipo:**
Para cada Responsabilidad:	
Nombre:	__init__(self, idregla, esquema, tabla, campo, operador, valor, idbasedato)
Descripción:	Constructor de la clase.
Nombre:	insertar(self, objconexion)
Descripción:	Inserta una regla en la BD.
Nombre:	modificar(self, objconexion)
Descripción:	Modifica una regla en la BD.
Nombre:	eliminar(self, objconexion)
Descripción:	Elimina una regla en la BD.

Tabla 34 Descripción de la Clase NRegla.

Nombre: NListaRegla	
Tipo de Clase: Negocio	
Atributo:	**Tipo:**
Para cada Responsabilidad:	
Nombre:	__init__(self)
Descripción:	Constructor de la clase.
Nombre:	cargar_todo(self, id, objconexion)
Descripción:	Carga todos los campos de la tabla Reglas.

Tabla 35 Descripción de la Clase NListaRegla.

Nombre: NReporte	
Tipo de Clase: Negocio	
Atributo:	**Tipo:**
Para cada Responsabilidad:	
Nombre:	__init__ (fecha, objconexion)
Descripción:	Constructor de la clase.
Nombre:	adicionar_rep(objconexion)
Descripción:	Adiciona el reporte.
Nombre:	modificar_rep(objconexion)
Descripción:	Modifica el reporte.
Nombre:	eliminar_rep(objconexion)
Descripción:	Elimina el reporte.
Nombre:	cargar_tipos()
Descripción:	Carga los reportes por tipo.
Nombre:	generar_reporte_por_filtro(filtro)
Descripción:	Genera el reporte por filtros.
Nombre:	verificar_reportadas(sentencias)
Descripción:	Verifica si las sentencias fueron reportadas.
Nombre:	cargar_sent_reportadas()
Descripción:	Carga las sentencias que fueron reportadas.
Nombre:	cargar_conf_reportadas()
Descripción:	Carga las confirmaciones que fueron reportadas.

Tabla 36 Descripción de la Clase NReporte.

3.3.5 Diagrama de Interacción.

Se utilizan los diagramas de interacción para modelar los aspectos dinámicos del sistema, lo que conlleva modelar instancias concretas de clases interfaces, componentes y nodos, junto con los mensajes enviados entre ellos, todo en el contexto de un escenario que ilustra un comportamiento. En el contexto de las clases se describe la forma en que grupos de objetos colaboran para proveer un comportamiento. Dentro de este tipo de diagramas están los de secuencia y los de colaboración. [24]

3.3.5.1 Diagrama de Secuencia.

Un diagrama de secuencia muestra las interacciones entre objetos ordenadas en secuencia temporal. Muestra los objetos que se encuentran en el escenario y la secuencia de mensajes intercambiados entre los objetos para llevar a cabo la funcionalidad descrita por el escenario.

Los diagramas de secuencia, formalmente diagramas de traza de eventos o de interacción de objetos, se utilizan con frecuencia para validar los casos de uso. Observando qué mensajes se envían a los objetos, componentes o casos de uso y viendo cuanto tiempo consume el método invocado, los diagramas de secuencia ayudan a comprender los cuellos de botella potenciales, para así poder eliminarlos. A la hora de documentar un diagrama de secuencia resulta importante mantener los enlaces de los mensajes a los métodos apropiados del diagrama de clases. [25]

A continuación se muestran los diagramas de secuencia del sistema divididos por casos de uso.

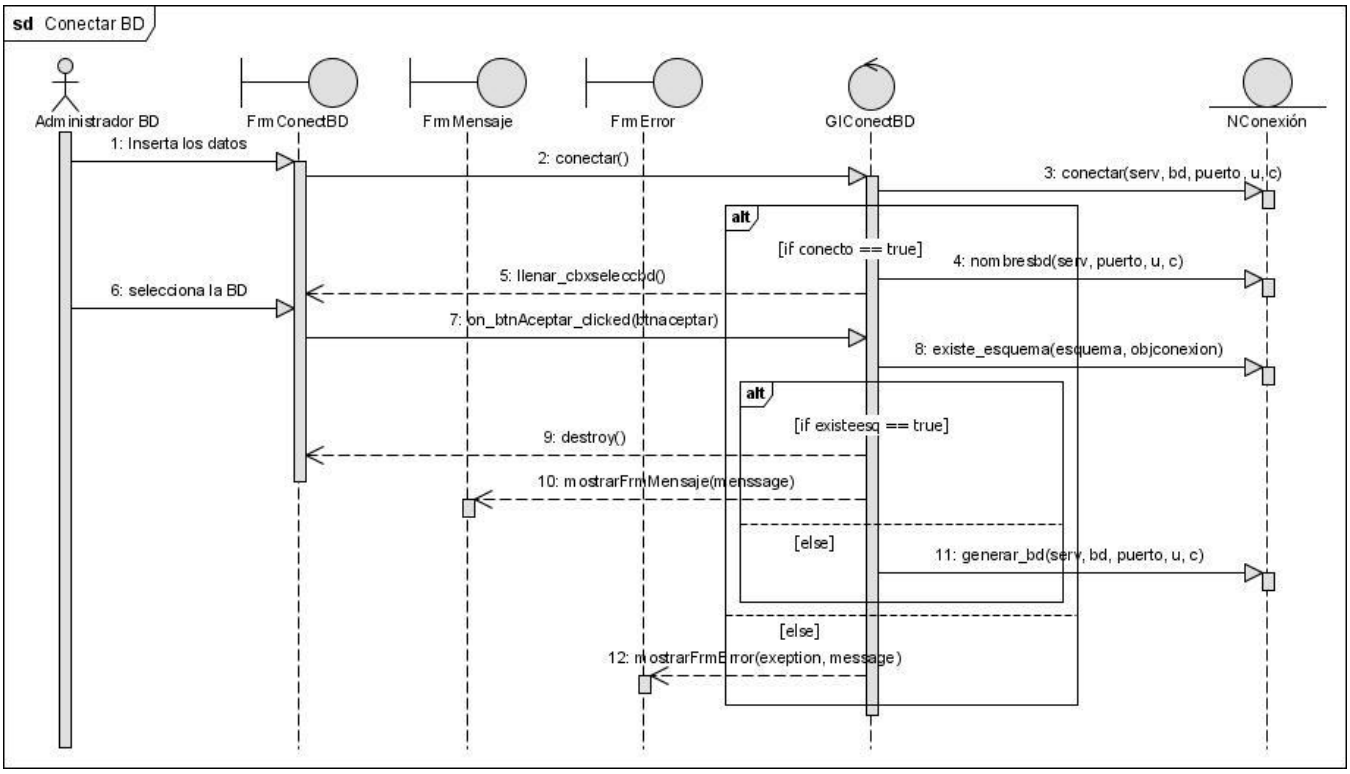


Fig. 30 Diagrama de Secuencia Caso de Uso Conectar BD.

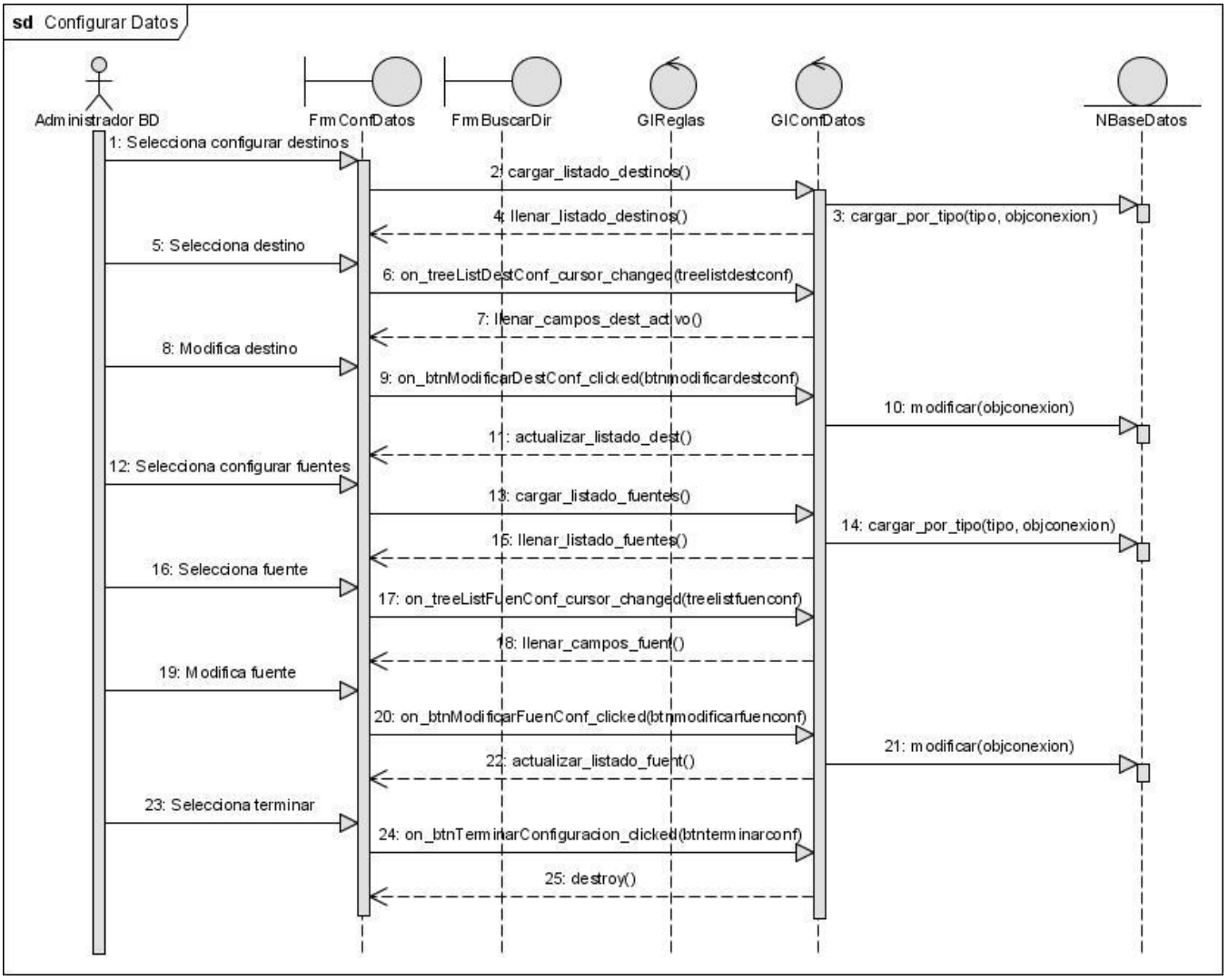


Fig. 31 Diagrama de Secuencia Caso de Uso Configurar Datos.

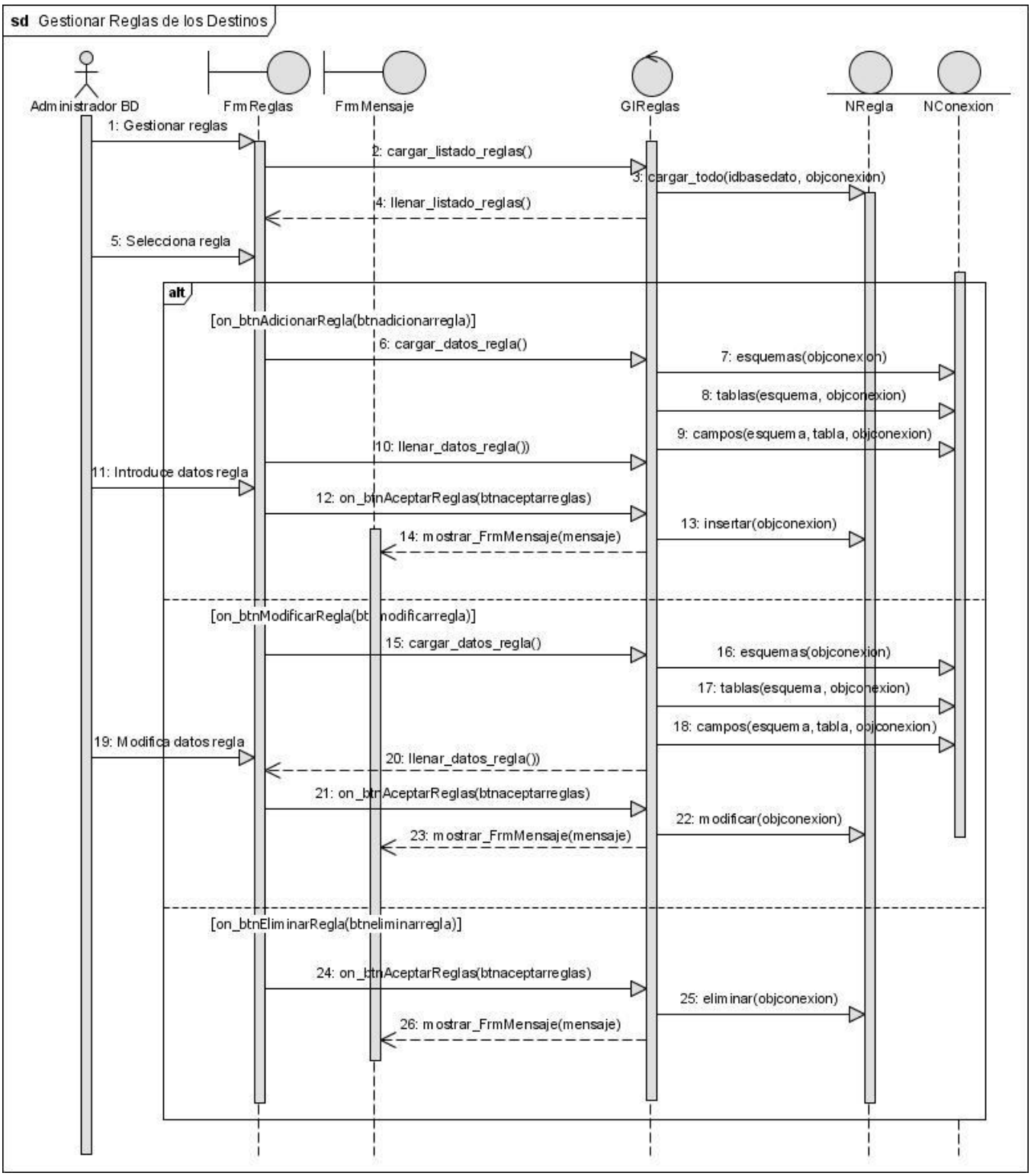


Fig. 32 Diagrama de Secuencia Caso de Uso Gestionar Reglas de los Destinos.

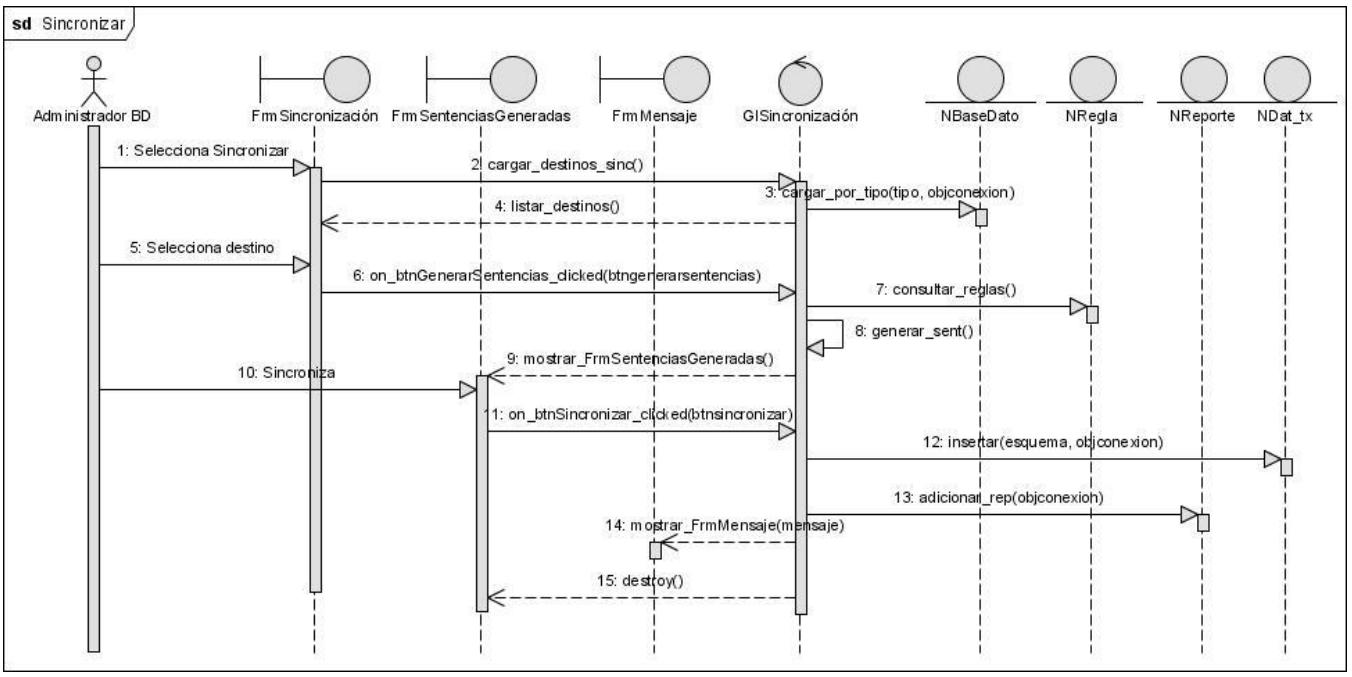


Fig. 33 Diagrama de Secuencia Caso de Uso Sincronizar.

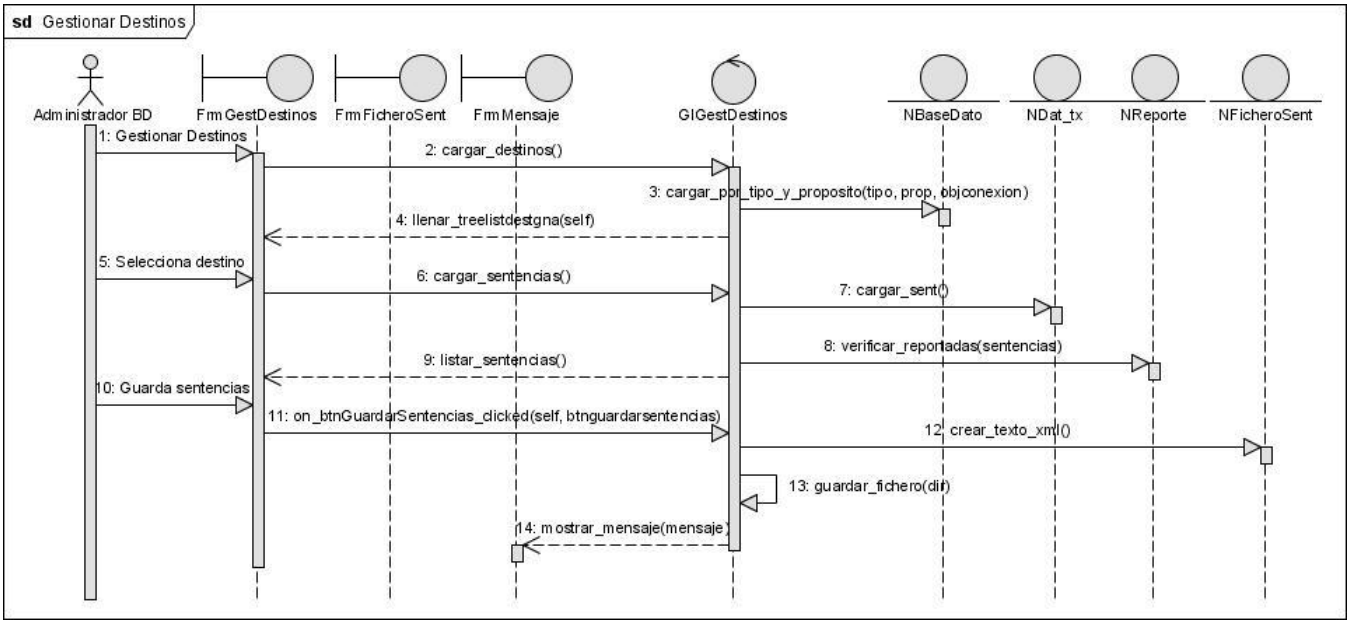


Fig. 34 Diagrama de Secuencia Caso de Uso Gestionar Destino de Datos.

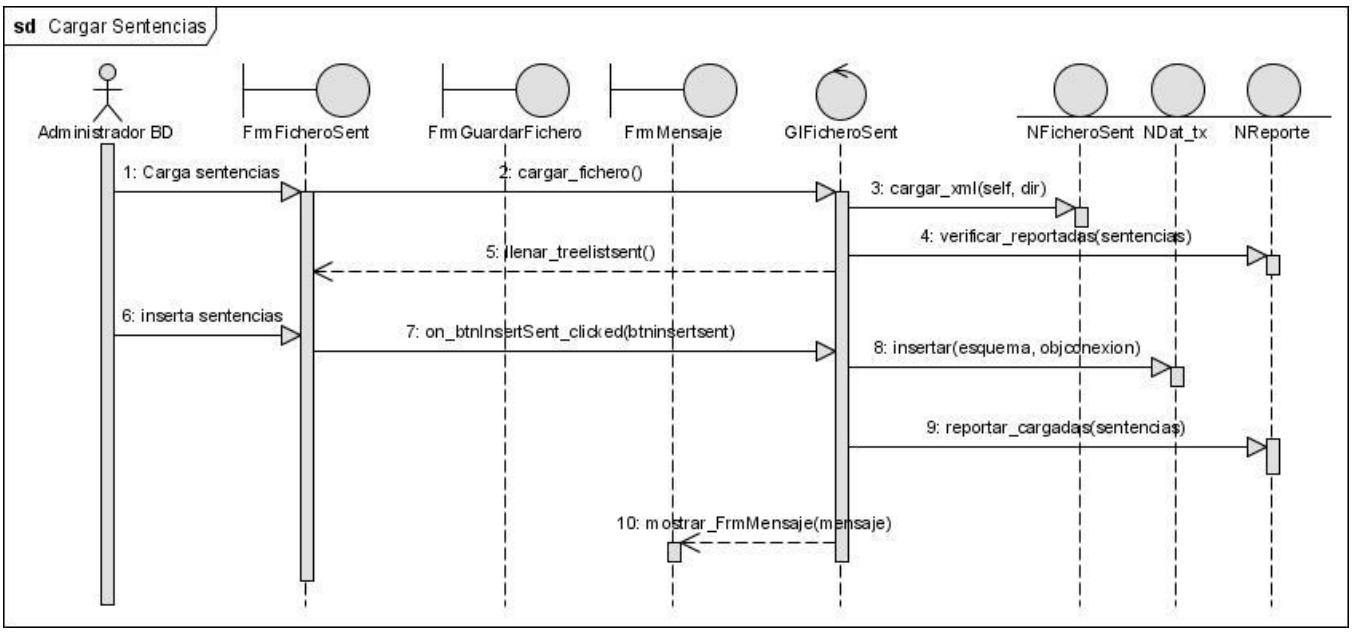


Fig. 35 Diagrama de Secuencia Caso de Uso Cargar Sentencias.

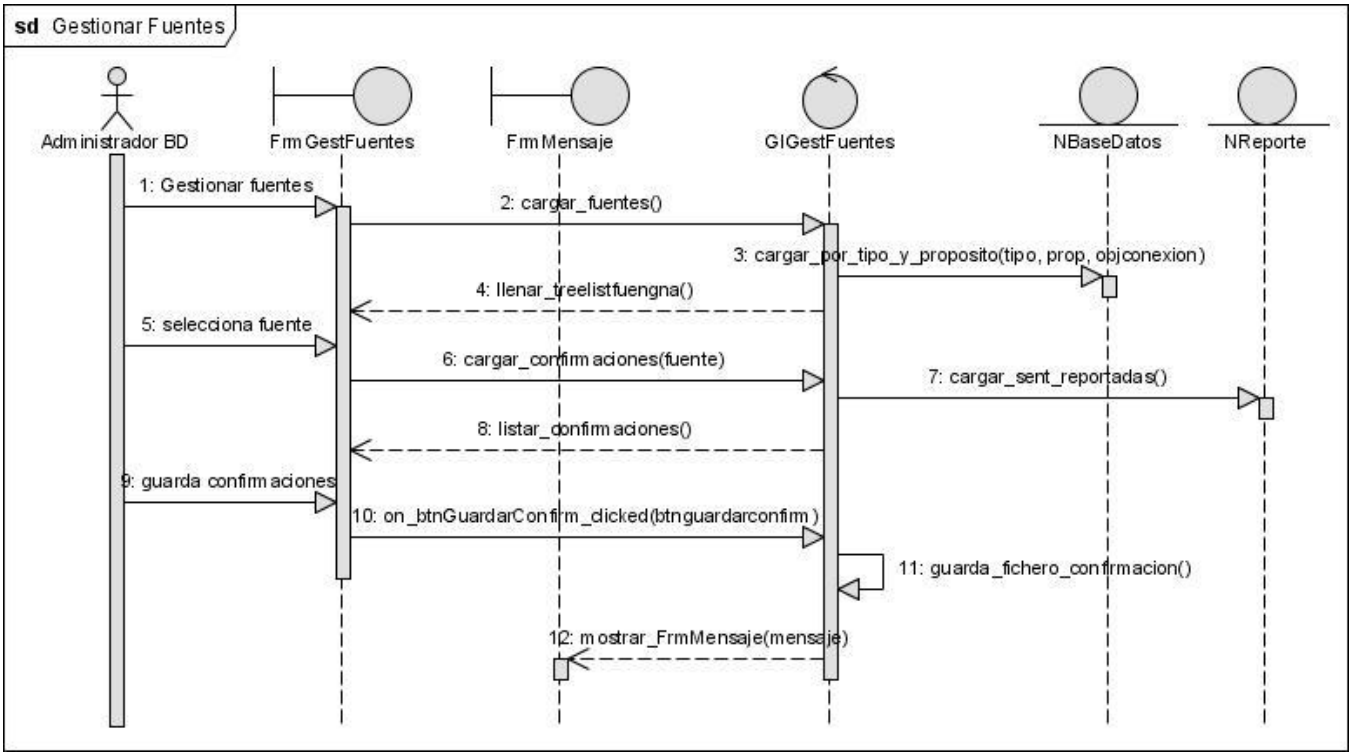


Fig. 36 Diagrama de Secuencia Caso de Uso Gestionar Fuente de Datos.

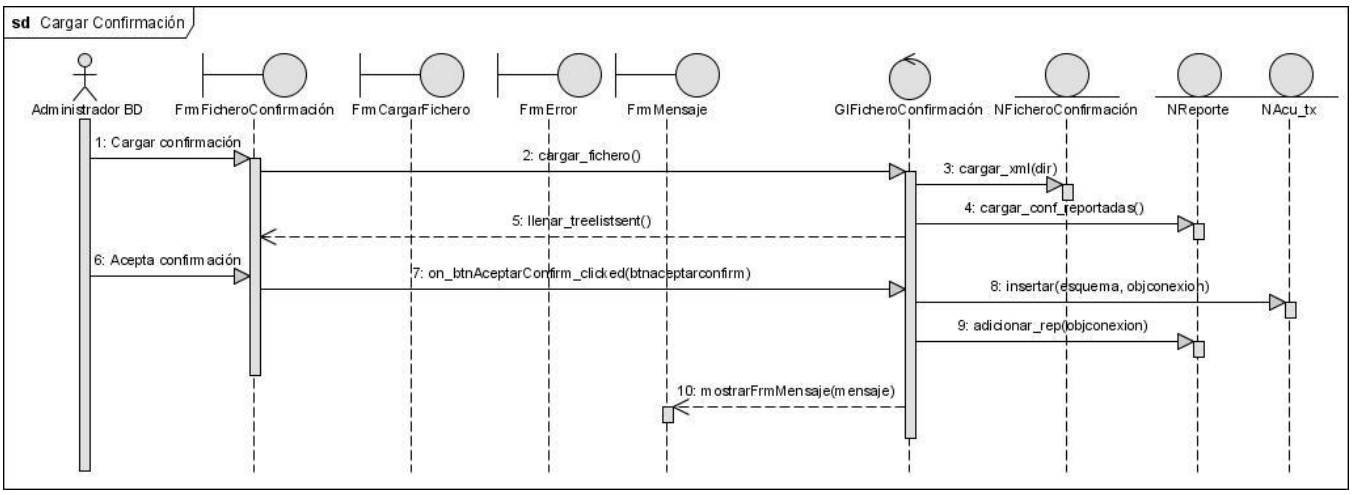


Fig. 37 Diagrama de Secuencia Caso de Uso Cargar Confirmación.

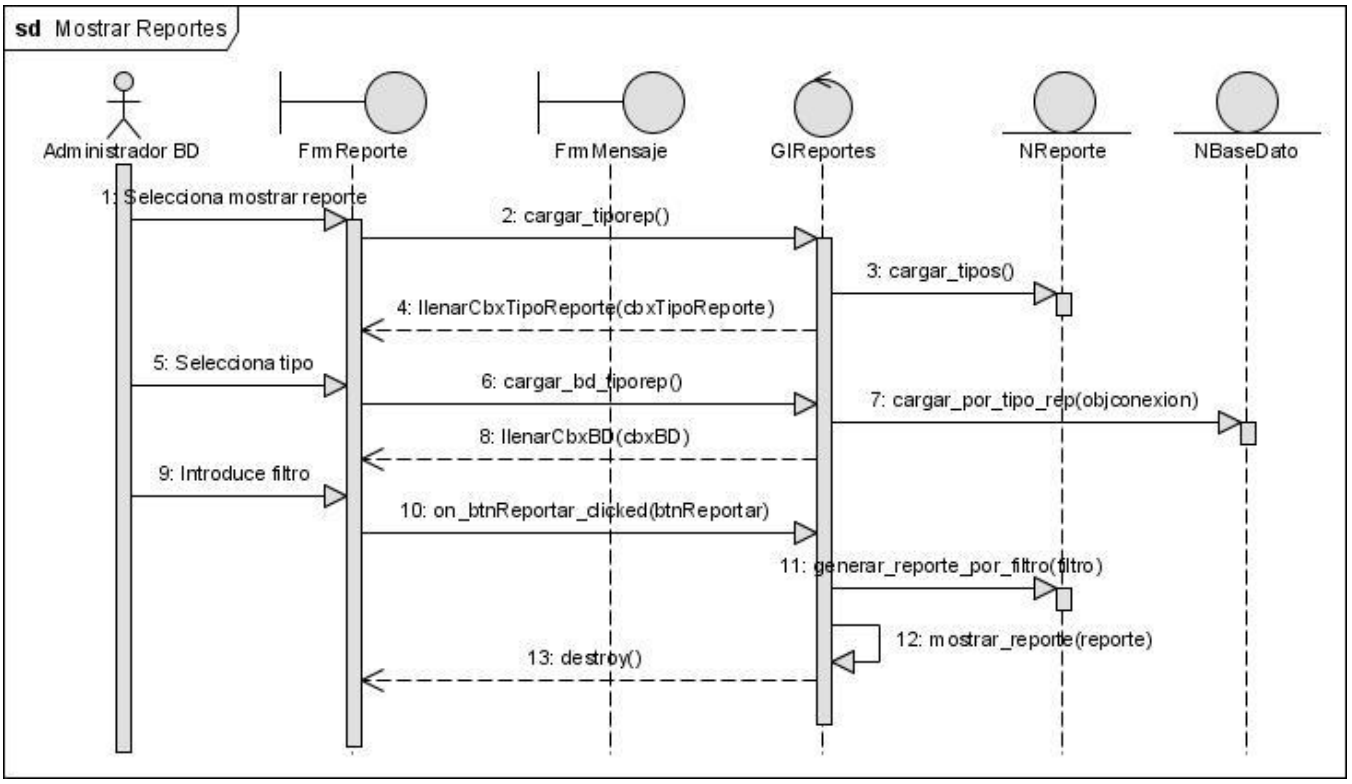


Fig. 38 Diagrama de Secuencia Caso de Uso: Mostrar Reportes.

3.3.6 Diseño de la Base de Datos.

3.3.6.1 Diagrama de Entidad Relación de la Base de Datos.

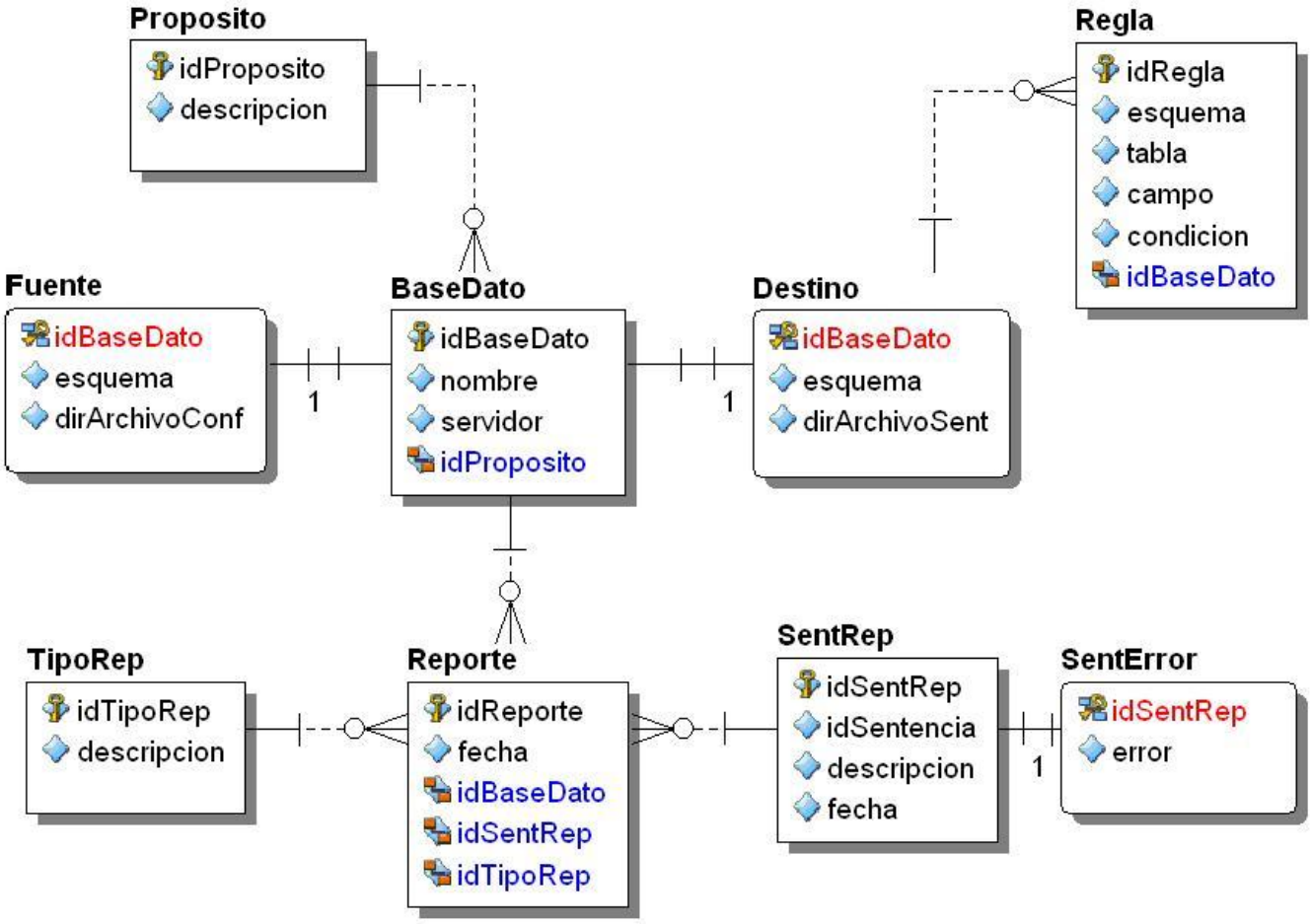


Fig. 39 Diagrama Entidad Relaciones BD.

3.3.6.2 Descripción de las Tablas.

Nombre: BaseDato		
Atributo	**Tipo**	**Descripción**
idBaseDato	auto numérico	Identificador de la Base de Datos.
nombre	text	Nombre de la Base de Datos.
Servidor	text	IP del Servidor.
idPropósito	int	Identificador del Propósito.

Tabla 37 Descripción de tabla BaseDato.

Nombre: Fuente.		
Atributo	**Tipo**	**Descripción**
idBaseDato	int	Identificador de la Base de Datos.
esquema	text	Esquema de la Fuente.
dirArchivoConf	text	Dirección donde se guardan los ficheros.

Tabla 38 Descripción de tabla Fuente.

Nombre: Destino		
Atributo	**Tipo**	**Descripción**
idBaseDato	int	Identificador de la Base de Datos.
esquema	text	Esquema del Destino.
dirArchivoSent	text	Dirección donde se guardan los ficheros.

Tabla 39 Descripción de tabla Destino.

Nombre: Propósito		
Atributo	**Tipo**	**Descripción**
idPropósito	auto numérico	Identificador del Propósito.
descripción	text	Tipo de propósito.

Tabla 40 Descripción de tabla Propósito.

Nombre: Regla		
Atributo	**Tipo**	**Descripción**
idRegla	auto numérico	Identificador de la Regla.
esquema	text	Esquema para la regla.
tabla	text	Tabla de la regla.
campo	text	Campo de la regla.
condición	text	Condición de la regla.
idBaseDato	interger	Identificador de la Base de Datos.

Tabla 41 Descripción de tabla Regla.

Nombre: SentRep		
Atributo	**Tipo**	**Descripción**
IdSentRep	auto numérico	Identificador de las sentencias reportadas.
idSentencia	interger	Identificador de la sentencia.
descripción	text	Descripción de la sentencia SQL.
fecha	timestamp	Fecha en que se reportó la sentencia.

Tabla 42 Descripción de tabla SentRep.

Nombre: SentError		
Atributo	**Tipo**	**Descripción**
idSentencia	interger	Identificador de la sentencia.
error	text	Descripción de la sentencia SQL.

Tabla 43 Descripción de tabla SentError.

Nombre: Reporte		
Atributo	**Tipo**	**Descripción**
idReporte	auto numérico	Identificador del Reporte.
fecha	text	Fecha del reporte.
idBaseDato	int	Identificador de la BD
idSentencia	int	Identificador de la sentencia.
idTipoRep	int	Identificador del tipo de reporte.

Tabla 44 Descripción de tabla Reporte.

Nombre: TipoRep		
Atributo	**Tipo**	**Descripción**
idTipoRep	int	Identificador del tipo de reporte.
descripción	text	Tipo de propósito.

Tabla 45 Descripción de tabla TipoRep.

3.4 Conclusiones.

En este capítulo se ha presentado el flujo de trabajo de más peso en la fase de Elaboración que propone la metodología Rational Unified Process (RUP) donde se han visto de una forma muy detallada los diagramas de clases del análisis y del diseño correspondientes. Además se presentaron los diagramas de interacción, específicamente de secuencia. Se definió el diagrama entidad-relación y las responsabilidades de cada uno de sus entidades y además fueron descritos los principios de diseño en los que se basa la solución propuesta.

CAPÍTULO 4: CONSTRUCCIÓN DE LA SOLUCIÓN PROPUESTA.

4.1 Introducción.

En este capítulo se presenta cómo se va a desarrollar la aplicación y cómo están distribuidos sus componentes a través del diagrama de componentes. Se describe el diagrama de despliegue así como la descripción de los casos de prueba para asegurar una óptima calidad del producto del software.

4.2 Implementación.

El Modelo de Implementación representa la composición física de la implementación en términos de los Subsistemas de Implementación, y los Elementos de Implementación (los directorios y los archivos, incluyendo código fuente, los datos, y los archivos ejecutables). Denota la implementación actual del sistema en términos de componentes y subsistemas de implementación. [26]

4.2.1 Diagrama de Despliegue.

El Diagrama de Despliegue muestra la disposición física de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Un nodo puede contener instancias de componentes software, objetos, procesos (caso particular de un objeto). [27]

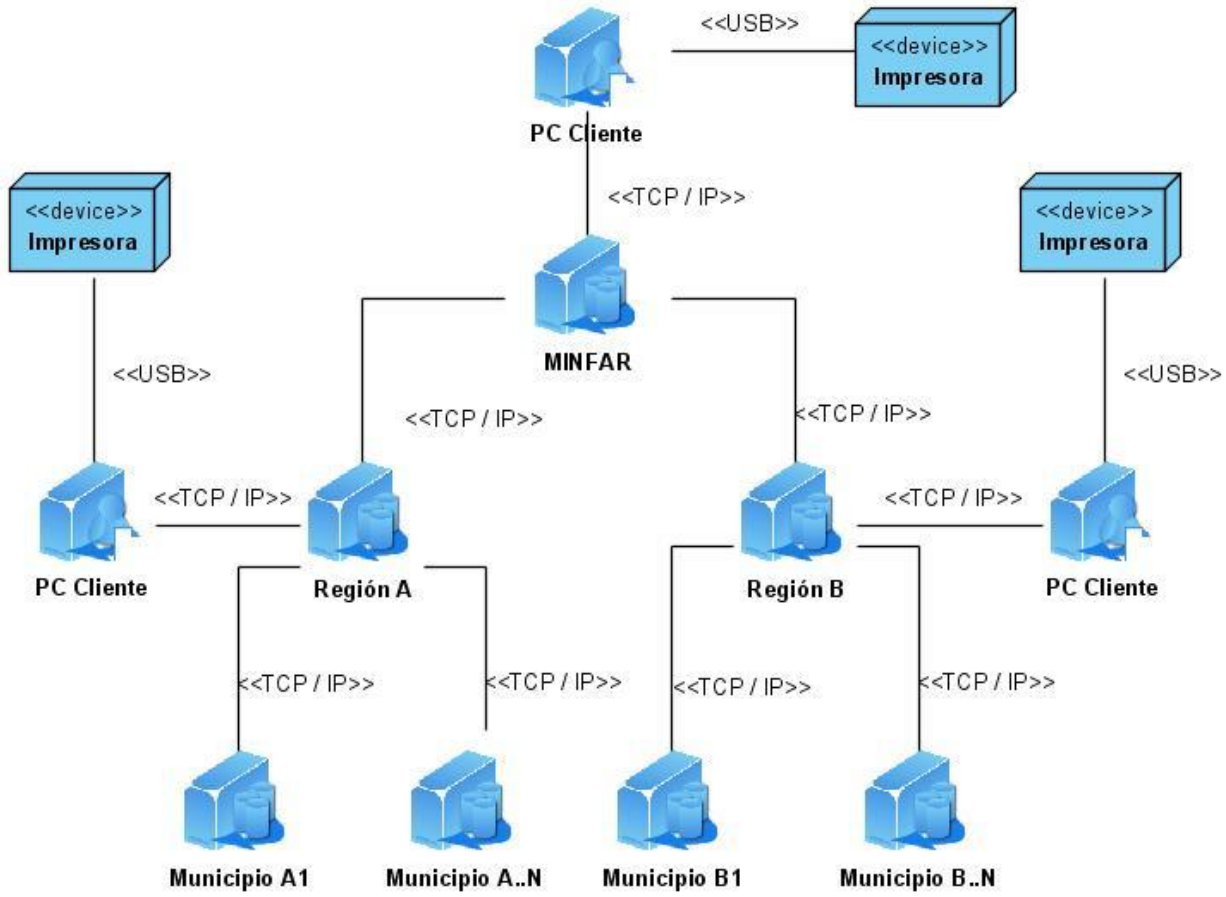


Fig. 40 Diagrama de Despliegue del Sistema.

El diagrama de despliegue se modela en dependencia del esquema jerárquico existente en las FAR que se explica en la introducción del trabajo de diploma. Cada entidad independientemente del nivel que represente contará con un servidor de datos y una terminal para el cliente, así como una impresora para imprimir los reportes que en su momento son exigidos y analizados por los especialistas y administradores del sistema. En su estructura cuenta con un grupo de regiones y municipios que responden al órgano del MINFAR. Dichas entidades se encuentran entrelazadas por una red TCP / IP por la cual se trasladan los datos entre los distintos sistemas de BD, que en una situación extraordinaria pudiera encontrarse en un estado aislado y no contar con este enlace. El sistema en su vista de despliegue mantendría sus funcionalidades al presentarse dicha situación extrema.

4.2.2 Diagrama de Componentes.

El diagrama de componentes se representa como un grafo donde los componentes del sistema se encuentran relacionados formando dependencias entre ellos. Entre los componentes se encuentran las

clases, interfaces, librerías y componentes ejecutables, donde algunos están dentro de paquetes de funcionalidades. Normalmente los diagramas de componentes se utilizan para modelar código fuente, versiones ejecutables, bases de datos físicas, entre otros. En fin, este diagrama muestra un conjunto de ficheros relacionados entre si para logar una completa funcionalidad del sistema. [28]

A continuación se muestra el diagrama de componentes del sistema:

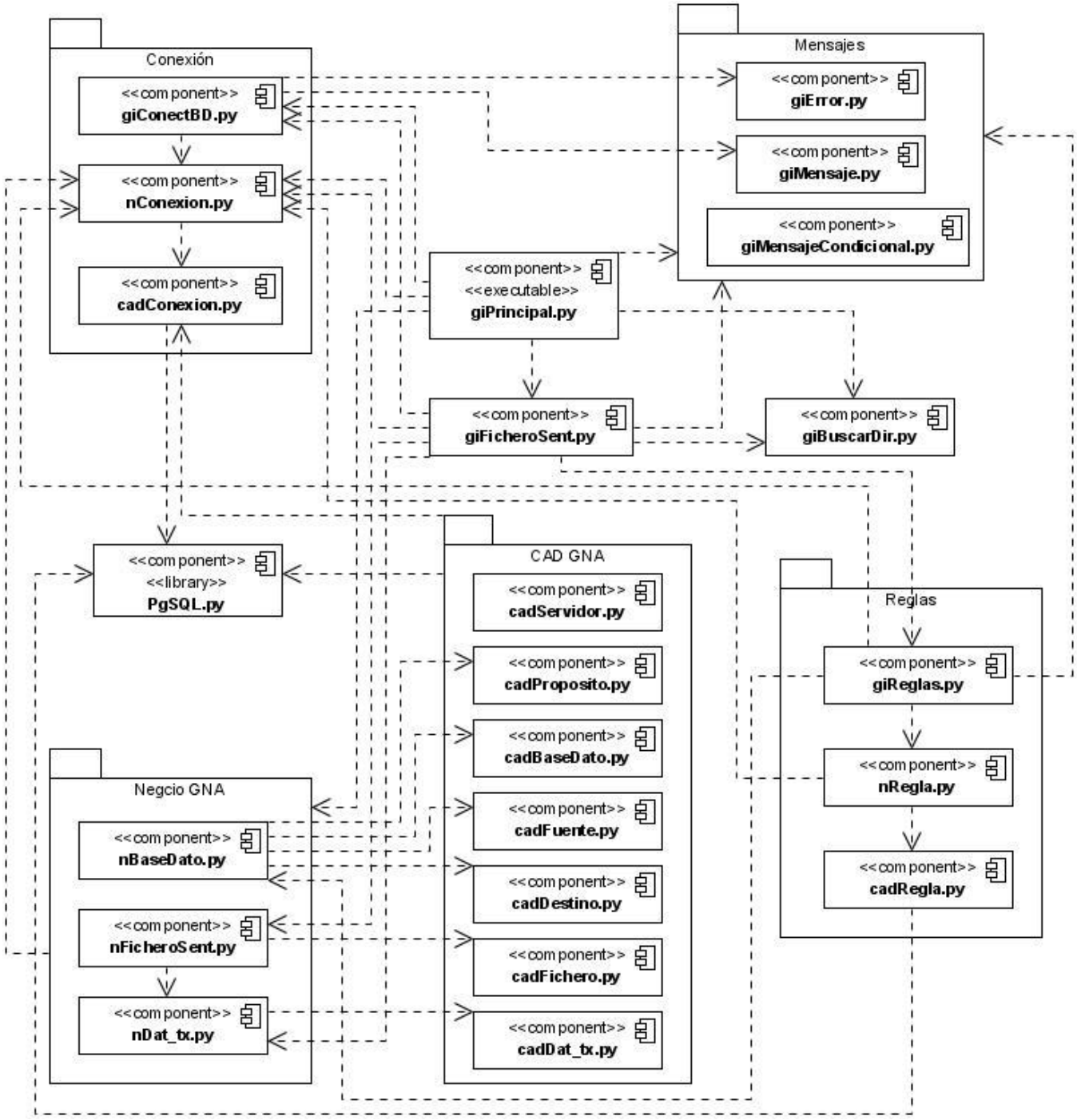


Fig. 41 Diagrama de Componentes del Sistema.

En el anterior diagrama se muestran cinco grupos de componentes agrupados por el tipo, la funcionalidad o el rol que juegan en el sistema. El paquete de conexión agrupa el conjunto de componentes que manejan las funciones para habilitar la conexión del sistema con la BD. Por otra parte los componentes de la Gestión de Nodos Aislados (GNA) se encontraran divididos en dos grupos de componentes: Negocio GNA referente a los componentes que conforman el negocio y CAD GNA referente a los componentes que conforman la capa de acceso a datos. Los componentes del tipo capa de acceso a datos van a depender directamente de la librería PgSQL.py para poder mapear los datos existentes en la capa de datos. Todos los componentes del negocio y gestión de interfaz serán dependencias del componente ejecutable, en este caso, giPrincipal.py que contiene el manejo de los eventos de la interfaz principal así como las funciones que controlan el sistema en general. El componente giFicheroSent.py controla todas las funciones referentes a los ficheros XML que serán creados y cargados con las sentencias SQL que serán transmitidas. Este fichero se guardara en directorios que serán manejados por el componente giBuscarDir.py. Los componentes referentes al manejo de los distintos mensajes se encuentran agrupados en el paquete Mensaje. Otro de los paquetes que se modelan en el diagrama de componentes es el de Reglas, agrupando los componentes del sistema que gestionan estas funciones.

4.3 Modelo de Prueba.

Es un proceso que corre en paralelo al proceso de desarrollo de software, y que se realiza por el convencimiento de que todo sistema debe ser “revisado” con el objetivo de establecer el nivel de calidad requerido. El modelo de prueba debe llevarse a cabo de una manera sistemática, y debe respaldarse en métricas bien definidas. Además genera información muy valiosa acerca de la madurez del producto de software que permite tomar decisiones importantes posteriormente.

Aplicando pruebas de software, se puede reducir el costo y el tiempo global del desarrollo, pues se detectan una buena cantidad de problemas en fases tempranas del proceso. [29]

A continuación se describen los modelos de pruebas confeccionados para los distintos casos de uso del sistema:

Caso de Uso: Conectar BD.

Sección.	**Entrada.**	**Resultados de la Prueba.**	**Condiciones.**
SC1: Flujo Básico.	Servidor(correcto) Puerto(correcto) Usuario(correcto) Contraseña(correcto)	Lista las bases de datos de prueba en el servidor local.	Existe el Servidor de BD.

	Servidor(correcto) Puerto(correcto) Usuario(correcto) Contraseña(correcto)	El sistema lanza una excepción mostrando un mensaje de error.	No existe el Servidor de BD especificado.
	Servidor(incorrecto) Puerto(correcto) Usuario(correcto) Contraseña(correcto)	El sistema lanza una excepción mostrando un mensaje de error.	
	Servidor(correcto) Puerto(incorrecto) Usuario(correcto) Contraseña(correcto)	El sistema lanza una excepción mostrando un mensaje de error.	
	Servidor(correcto) Puerto(correcto) Usuario(incorrecto) Contraseña(incorrecto)	El sistema lanza una excepción mostrando un mensaje de error.	

Tabla 46 Descripción de Prueba para el CU Conectar BD.

Caso de Uso: Configurar Datos.

Sección.	**Entrada.**	**Resultados de la Prueba.**	**Condiciones.**
SC2: Flujo Básico	Se introducen los datos de un destino nuevo correctamente y se presiona el botón “Aceptar”.	Se actualizan los datos del destino seleccionado.	
	Se modifican los datos de un destino existente y se presiona el botón “Aceptar”.	Se actualizan los datos del destino seleccionado.	Existen al menos un destino en el listado.
	Se elimina un destino de la lista de destinos.	Se actualizan los datos del destino seleccionado.	Existen al menos un destino en el listado.
	Se modifica un destino y	No se produce ningún	Existen al menos un

	se presiona el botón “Cancelar”.	resultado.	destino en el listado.
SC3: Modificar Fuentes.	Se introducen los datos de una fuente nueva correctamente y se presiona el botón “Aceptar”.	Se actualizan los datos de la fuente seleccionada.	
	Se modifica un dato de una de las fuentes existentes correctamente y se presiona el botón “Aceptar”.	Se actualizan los datos de la fuente seleccionada.	Existen al menos una fuente en el listado.
	Se elimina una fuente del listado de fuente.	Se actualizan los datos de la fuente seleccionada.	Existen al menos una fuente en el listado.
	Se modifica una fuente del listado y se presiona el botón “Cancelar”.	No se produce ningún resultado.	Existen al menos una fuente en el listado.

Tabla 47 Descripción de Prueba para el CU Configurar Datos.

Caso de Uso: Gestionar Regla de los Destinos.

Sección.	**Entrada.**	**Resultados de la Prueba.**	**Condiciones.**
SC4: Flujo Básico.	Se entran los datos correctos y se presiona el botón “Aceptar”.	Se adiciona la regla descrita en el destino antes seleccionado.	Seleccionar un elemento en el listado de destinos.
	Se entran los datos correctos y se presiona el botón “Cancelar”.	No se adiciona la regla descrita en el destino antes seleccionado.	Seleccionar un elemento en el listado de destinos.
	Se entran los datos incorrectos y se presiona el botón “Aceptar”.	Se muestra en mensaje de error especificando los campos incorrectos.	Seleccionar un elemento en el listado de destinos.
	Se entran los datos incorrectos y se presiona el botón “Cancelar”.	No se adiciona la regla descrita en el destino antes seleccionado.	Seleccionar un elemento en el listado de destinos.

Tabla 48 Descripción de Prueba para el CU Gestionar Regla de los Destinos.

Caso de Uso: Sincronizar.

Sección.	Entrada.	Resultados de la Prueba.	Condiciones.
SC5: Flujo Básico.	Se selecciona un destino y presiona el botón “Generar Sentencias”.	Se muestra una ventana con las sentencias generadas.	Existen al menos un destino en el listado.
	No se selecciona ningún destino y se presiona el botón “Generar Sentencias”.	Se muestra un mensaje de error: “No existe ningún destino seleccionado”	
	Se presiona el botón “Aceptar”.	Quedan guardadas las sentencias en la tabla de transmisión	Están generadas las sentencias para el destino seleccionado.
	Se presiona el botón “Cancelar”.	No se guardan las sentencias. No ocurre ningún cambio en el sistema.	

Tabla 49 Descripción de Prueba para el CU Sincronizar.

Caso de Uso: Guardar Sentencias.

Sección.	Entrada.	Resultados de la Prueba.	Condiciones.
SC6: Flujo Básico.	Se selecciona un destino y se presiona el botón “Guardar Sentencias”	Se muestra un diálogo de búsqueda para seleccionar el directorio para guardar el fichero.	Existen al menos un destino en el listado.
	No se selecciona ningún destino y se presiona el botón “guardar sentencias”	Se muestra un mensaje de error: “No existe ningún destino seleccionado”.	
	Se presiona el botón “Aceptar” del dialogo de búsqueda	Se guardan las sentencias en el fichero Consulta.	Seleccionar un directorio.
	Se presiona el botón “Cancelar” del dialogo de	No se guardan las sentencias en el fichero.	

	búsqueda		

Tabla 50 Descripción de Prueba para el CU Guardar Sentencias.

Caso de Uso: Cargar Sentencias.

Sección.	Entrada.	Resultados de la Prueba.	Condiciones.
SC7: Flujo Básico.	Se escoge un fichero correcto en el dialogo de búsqueda y se presiona "Aceptar".	El sistema muestra un listado de sentencias a ejecutar.	Seleccionar un fichero en el directorio.
	Se escoge un fichero incorrecto en el dialogo de búsqueda y se presiona "Aceptar".	El sistema muestra un mensaje de error "Fichero Incorrecto".	Seleccionar un fichero en el directorio.
	Se escoge un fichero correcto en el dialogo de búsqueda y se presiona "Cancelar".	No se carga el fichero de sentencias.	Seleccionar un fichero en el directorio.
	Se presiona el botón "Aceptar Envío".	Se ejecutan las sentencias y se actualiza el sistema.	Están listadas las sentencias del fichero.
	Se presiona el botón "Cancelar Envío".	No se ejecutan las sentencias y no se produce ningún cambio en el sistema.	Están listadas las sentencias del fichero.

Tabla 51 Descripción de Prueba para el CU Cargar Sentencias.

Caso de Uso: Guardar Confirmaciones.

Sección.	Entrada.	Resultados de la Prueba.	Condiciones.
SC8: Flujo Básico.	Se escogen una fuente y se presiona el botón de "Guardar Confirmación".	Se muestra un dialogo para escoger el sitio para guardar el fichero.	Están listados los ID de la fuente seleccionada.
	No se escoge ninguna fuente y se presiona el botón de "Guardar	Se muestra un mensaje de error "No existe ninguna fuente seleccionada".	

	Confirmación".		
	Se presiona el botón "Aceptar" del dialogo de búsqueda.	Se guardan los datos en un fichero listos para ser transmitidos por vía manual.	Seleccionar un directorio.
	Se presiona el botón "Cancelar" del dialogo de búsqueda.	No se guardan los datos.	

Tabla 52 Descripción de Prueba para el CU Guardar Confirmaciones.

Caso de Uso: Cargar Confirmaciones.

Sección.	**Entrada.**	**Resultados de la Prueba.**	**Condiciones.**
SC9: Flujo Básico.	Se presiona el botón "Importar Confirmación".	Se muestra en dialogo de búsqueda para seleccionar el fichero de Acuse.	Seleccionar un fichero en el directorio.
	Se escoge el fichero de Acuse correcto y se presiona el botón "Aceptar".	Se muestra un listado con los ID de las sentencias que fueron ejecutadas en el destino.	Seleccionar un fichero en el directorio.
	Se presiona el botón "Cancelar".	No se realiza ningún cambio en el sistema.	
	Se escoge el fichero de Acuse incorrecto y se presiona el botón "Aceptar".	El sistema muestra un mensaje de error: "Fichero Incorrecto".	Seleccionar un fichero en el directorio.
	Se presiona el botón "Aceptar Confirmación".	Se ejecutan los datos del fichero acuse y quedan actualizados los destinos en la BD.	Están listados los datos del fichero seleccionado.
	Se presiona el botón "Cancelar Confirmación".	No se ejecutan los datos del fichero acuse en la BD.	Están listados los datos del fichero seleccionado.

Tabla 53 Descripción de Prueba para el CU Cargar Confirmaciones.

Caso de Uso: Mostrar Reportes.

Sección.	Entrada.	Resultados de la Prueba.	Condiciones.
SC10: Flujo Básico.	Se escoge la opción de mostrar reporte de "Sentencias Recepcionadas" y se presiona el botón de Mostrar Reporte.	Se muestra un reporte con los parámetros escogidos.	Conexión habilitada a un servidor DB, específicamente a una BD.
	Se escoge la opción de mostrar reporte de "Confirmaciones Transmitidas" y se presiona el botón de Mostrar Reporte.	Se muestra un reporte con los parámetros escogidos.	Conexión habilitada a un servidor DB, específicamente a una BD.
	Se escoge la opción de mostrar reporte de "Sentencias Sincronizadas" y se presiona el botón de Mostrar Reporte.	Se muestra un reporte con los parámetros escogidos.	Conexión habilitada a un servidor DB, específicamente a una BD.
	Se escoge la opción de mostrar reporte de "Sentencias Transmitidas" y se presiona el botón de Mostrar Reporte.	Se muestra un reporte con los parámetros escogidos.	Conexión habilitada a un servidor DB, específicamente a una BD.
	Se escoge la opción de mostrar reporte de "Confirmaciones Recepcionadas" y se presiona el botón de Mostrar Reporte.	Se muestra un reporte con los parámetros escogidos.	Conexión habilitada a un servidor DB, específicamente a una BD.

Tabla 54 Descripción de Prueba para el CU Mostrar Reportes.

4.4 Conclusiones.

En el capítulo se abordó la forma en que se desarrolló la aplicación haciendo uso para ello del diagrama de componentes. Además se realizó una descripción de la forma de despliegue del sistema mediante el diagrama de despliegue. Por último fueron realizados los diferentes casos de prueba con los que se midió la calidad del sistema construido.

CONCLUSIONES

En este trabajo de diploma se han conceptualizado las reglas de los procesos informáticos dentro de las FAR, específicamente los referidos a los procesos de réplica de los Sistemas de Bases de Datos Distribuidas con que cuenta.

Se ha hecho énfasis en el estudio de herramientas para la réplica y sincronización que se usan en sistemas donde se manejan los datos con el gestor PostgreSQL, incluyendo los usados por las FAR en la actualidad, resaltando sus potenciales y debilidades para un análisis completo.

A raíz del estudio de las herramientas existentes se han seleccionado las más óptimas dentro de las usadas por el ministerio, pertenecientes al mundo del software libre, para el diseño y la implementación de este sistema.

Se usó RUP como proceso de desarrollo de software a seguir para la construcción de la solución planteada por el análisis, así como Visual Paradigm 3.0 Suite Edition como herramienta para la construcción de los diagramas UML de los distintos modelos existentes en el proceso, en los cuales se encuentra el Modelo de Dominio escogido por las características y conceptos del sistema.

Se usaron las herramientas establecidas por las FAR para el desarrollo de los componentes del sistema tales como el lenguaje de programación Python en su versión 2.5, y para el desarrollo de los componentes visuales Glade con las librerías GTK+ de interfaz de Usuario para mostrar un entorno visual de interfaces amigables para los operadores del sistema.

Se implemento el sistema con resultados satisfactorios en las pruebas realizadas alcanzando los objetivos predefinidos.

Por tanto se consideran cumplidos los objetivos y tareas trazadas en el inicio de este trabajo, donde se materializó la hipótesis planteada demostrándose que con el diseño y realización de una aplicación de escritorio capaz de sincronizar de forma inicial las bases de datos con gestores PostgreSQL y con funcionalidades para la manipulación de nodos independientemente del medio de comunicación, se podrá conseguir la sincronización inicial de las bases de datos distribuidas de las FAR y la gestión de los nodos aislados por no conectividad, cumpliendo con las normas establecidas entre los servidores de datos de las FAR.

RECOMENDACIONES

Este trabajo propone y construye un sistema para el manejo de sistemas de bases de datos dentro de las FAR donde se manejan las BD con el gestor PostgreSQL y se usa una plataforma GNU / Linux por lo que se recomienda proseguir el trabajo en el ámbito migratorio para poder ser usado por aplicaciones en distintas plataformas como Windows y Mac OS entre otros, manteniendo el gestor de BD para el que fue concebido en este caso PostgreSQL, el cual soporta varias plataformas.

Se recomienda además continuar investigando las herramientas que surgen al paso de los días y el desarrollo de la tecnología para enriquecer el tema del manejo de BD, y complementar el sistema con estos avances en versiones posteriores.

Este sistema es capaz de ejecutar las acciones de réplica bajo condiciones extremas como la no existencia de una conexión de red entre dos servidores, por lo que se recomienda este sistema como solución para la exportación e importación de datos, garantizando una integridad y actualidad de sus datos.

Se recomienda extender la solución de sincronización no solo en condiciones donde no existan datos en el destino (sincronización inicial), sino también para aquellos casos donde la fuente y el destino trabajan sin ningún nivel de intercambio y es necesario comenzar un proceso de intercambio de los datos.

BIBLIOGRAFÍA

CONSULTADA:

1. Rolando Alfredo Hernández León y Sayda Coello González. *El Paradigma Cuantitativo de la Investigación Científica.Ciudad de la Habana. 2002.* ISBN 959-16-0343-6.

2. **Korth, Henry F. y Silberschatz, Abraham. Ed. McGraw-Hill.** *Fundamentos de bases de datos.* 2002. ISBN 8448136543.

3. **Prentice, C.J. Ed.** *Introducción a los Sistemas de Bases de datos.* 2001. ISBN 968-444-419-2.

4. Modelo de Dominio.
[Consultado: abril 25, 2008.]
http://lsi.ugr.es/~ig1/isoo/larman/Modelo%20del%20dominio.pdf

5. **Ivar Jacobson, Grady Booch y James Rumbaugh.** *El Porceso Unificado de Desarrollo de Software.* 2000. ISBN 84-7829-036-2.

CITADA:

1. Fundamentos. *e-Quality.* [En línea] [Citado el: 21 de mayo de 2008.]
http://www.e-quallity.net/definiciones.php

2. Diagrama de Componentes. *G.R.I.S. Grupo de Redes e Ingeniería del Software.* [En línea] [Citado el: 16 de mayo de 2008.]
http://www-gris.det.uvigo.es/~avilas/UML/node49.html

3. Diagrama de Despliegue. *G.R.I.S. Grupo de Redes e Ingeniería del Software.* [En línea] [Citado el: 16 de mayo de 2008.]
http://www-gris.det.uvigo.es/~avilas/UML/node50.html

4. Modelo de Implementación. *Me Rinde.* [En línea] [Citado el: 16 de mayo de 2008.]
http://merinde.rinde.gob.ve/index.php?option=com_content&task=view&id=96&Itemid=297

5. Diagrama de Secuencia. *G.R.I.S. Grupo de Redes e Ingeniería del Software.* [En línea] [Citado el: 11 de mayo de 2008.]
http://www-gris.det.uvigo.es/~avilas/UML/node42.html

6. Diagramas de Interacción. *G.R.I.S. Grupo de Redes e Ingeniería del Software.* [En línea] [Citado el: 10 de mayo de 2008.]
http://www-gris.det.uvigo.es/~avilas/UML/node41.html

7. Diagrama de Clases del Análisis. *Ayuda Extendidad del Rational.* [En línea] 2003. [Citado el: 7 de mayo de 2008.]

8. Actores del Sistema. *Ayuda Extendidad del Rational.* [En línea] 2003. [Citado el: 6 de mayo de 2008.]

9. Capturar los Requerimientos no funcionales. *Facultad de Ingeniería - Universidad de Uruguay.* [En línea] [Citado el: 25 de abril de 2008.]
http://www.fing.edu.uy/inco/cursos/ingsoft/pis/memoria/experiencia2000/modelo_de_proceso1/lineas_de_trabajo/requerimientos/R10.htm

10. **Wiegers, Karl E.** *Software Requirements 2: Practical techniques for gathering and managing requirements throughout the product development cycle. s.l.* s.l. : Microsoft Press, 2003. 0-7356-1879-8.

11. Analisis y diseño orientado a objetos. *Una Herramienta CASE para ADOO: Visual Paradigm.* [En línea] [Citado el: 5 de marzo de 2008.]
http://alarcos.inf-cr.uclm.es/per/fgarcia/isoftware/doc/LabTr1_VP.pdf

12. Visual Paradigm for UML (ME) 6.0. *Sitio de Descarga de Software.* [En línea] [Citado el: 4 de marzo de 2008.]
http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(M%C3%8D)_14720_p/

13. *Visual Paradigm - Build Quality Applications Faster, Better and Cheaper.* [En línea] [Citado el: 4 de marzo de 2008.]
http://www.visual-paradigm.com

14. Introduction to OMG's Unified Modeling Language. *Object Management* . [En línea] 1997-2008. [Citado el: 16 de febrero de 2008.]
http://www.omg.org/gettingstarted/what_is_uml.htm

15. **Jacobson, Ivar, Booch, Grady and Rumbaugh, James.** *El Porceso Unificado de Desarrollo de Software.* 2000. 84-7829-036-2.

16. Glade - a User Interface Designer for GTK+ and GNOME. *Glade.* [En línea] 2007. [Citado el: 16 de febrero de 2008.]
http://glade.gnome.org/index.html

17. PyGTK: GTK+ for Python. *GTK Sitio Oficial.* [En línea] 4 de Julio de 2004-2006. [Citado el: 16 de febrero de 2008.]
http://www.pygtk.org/

18. What is GTK+? *The GTK Project.* [En línea] 2007-2008. [Citado el: 16 de febrero de 2008.]
http://www.gtk.org/

19. What is Python? *Python Programming Language -- Official Website.* [En línea] 1990-2007. [Citado el: 15 de febrero de 2008.]
http://www.python.org

20. PgCluster. *PgCluster.* [En línea] [Citado el: 13 de febrero de 2008.]
http://pgcluster.projects.postgresql.org/index.html

21. Slony-I. *Slony-I Enterprise-level replication sistem.* [En línea] 2007. [Citado el: 15 de febrero de 2008.]

http://slony.info

22. PostgreSQL. *PostgreSQL Global Development Group.* [En línea] 1996-2008. [Citado el: 15 de febrero de 2008.]
http://www.postgresql.org

23. Freeware. *Utilidades Utiles.* [En línea] 2003-2008. [Citado el: 4 de marzo de 2008.]
http://www.utilidades-utiles.com/descargar-freeware.html

24. Open Source. *Open Source Initiative.* [En línea] 24 de julio de 2006. [Citado el: marzo de 4 de 2008.]
http://www.opensource.org/docs/definition.html

25. ¿Qué es Copyleft? *GNU.* [En línea] 12 de abril de 2008. [Citado el: 28 de abril de 2008.]
http://www.gnu.org/copyleft/copyleft.es.html

26. Introducción a las licencias de software libre. *La Espiral.* [En línea] 16 de abril de 2002. [Citado el: 21 de febrero de 2008.]
http://www.laespiral.org/articulos/licencias/licencias.html

27. La Definición de Software Libre. *GNU Operating System.* [En línea] 21 de abril de 2008. [Citado el: 3 de mayo de 2008.]
http://www.gnu.org/philosophy/free-sw.es.html

28. Tipos de Fragmentación de Datos. *Departamento de Computacion de CINVESTAV.* [En línea] 2007. [Citado el: 20 de febrero de 2008.]
http://www.cs.cinvestav.mx/SC/prof_personal/adiaz/Disdb/Cap_3.html

29. INTERNET, HACIA EL ALMACENAMIENTO Y LA GESTIÓN DE DATOS. *Construcciones Digitales.* [En línea] [Citado el: 13 de febrero de 2008.]
http://www.construccionesdigitales.com/Internet/almacenamiento%20de%20datos%20en%20Internet.htm.

GLOSARIO

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

Artefactos: En términos de ingeniería de software, los artefactos son productos tangibles del proceso.

Bucle: En términos de programación, es una sentencia que se realiza repetidas veces a un trozo aislado de código, hasta que la condición asignada a dicho bucle deje de cumplirse. Generalmente, un bucle es utilizado para hacer una acción repetida sin tener que repetir varias veces el mismo código, lo que ahorra tiempo, deja el código más claro y facilita su modificación en el futuro.

Casos de Uso: En ingeniería del software, es una técnica para la captura de requisitos potenciales de un nuevo sistema o una actualización de software. Cada caso de uso proporciona uno o más escenarios que indican cómo debería interactuar el sistema con el usuario o con otro sistema para conseguir un objetivo específico.

Ciencias de la computación: Abarcan el estudio de las bases teóricas de la información y la computación y su aplicación en sistemas computacionales.

Clúster: En términos de Slony, un clúster son dos bases de datos PostgreSQL que intervienen en la réplica.

Código fuente: En programación, el texto escrito en un lenguaje de programación que ha de ser compilado o interpretado para ejecutarse en una computadora.

Configuración: Es un conjunto de datos que determina el valor de algunas variables de un programa o sistema de software, estás opciones generalmente son cargadas en su inicio y en algunos casos se deberá reiniciar para poder ver los cambios.

Conmutación: Es una técnica que sirve para hacer un uso eficiente de los enlaces físicos en una red de computadoras.

Consulta: En términos de base de datos, es un código donde se describe una acción a ejecutar.

Esquema: Un esquema es la definición de una estructura (generalmente relaciones o tablas de una base de datos), es decir, determina la identidad de la relación y que tipo de información podrá ser almacenada dentro de ella; en otras palabras: son los metadatos de la relación.

Flujo de trabajo (workflow en inglés): Es el estudio de los aspectos operacionales de una actividad de trabajo: cómo se estructuran las tareas, cómo se realizan, cuál es su orden correlativo, cómo se sincronizan, cómo fluye la información que soporta las tareas y cómo se le hace seguimiento al cumplimiento de las tareas.

Glosario de Términos

GNU: Proyecto con el objetivo de crear un sistema operativo completamente libre: el sistema GNU. Se establecieron algunas motivaciones para realizar el proyecto GNU, entre las que destaca "volver al espíritu de cooperación que prevaleció en los tiempos iníciales de la comunidad de usuarios de computadoras".

Hiperenlace o hipervínculo (enlace, vínculo, hipervínculo o link): Es un elemento de un documento electrónico que hace referencia a otro recurso.

Hipermedia: Es el término con que se designa al conjunto de métodos o procedimientos para escribir, diseñar, o componer contenidos que tengan texto, video, audio, mapas u otros medios, y que además tenga la posibilidad de interactuar con los usuarios.

Hipertexto: En informática, es el nombre que recibe el texto que, en la pantalla de una computadora, conduce al usuario a otro texto relacionado. La forma más habitual de hipertexto en documentos es la de hipervínculos o referencias cruzadas automáticas que van a otros documentos.

HTML (Lenguaje de Marcas de Hipertexto, HyperText Markup Language en Inglés): Es el lenguaje de marcado predominante para la construcción de páginas web.

IDE: Un **entorno de desarrollo integrado** o en inglés *Integrated Development Environment* (***'IDE'***) es un programa compuesto por un conjunto de herramientas para un programador.

Identificador: Nombran entidades. El concepto es análogo al de "nombre". Los identificadores se usan en prácticamente todos los sistemas de procesamiento de la información. Nombrar las entidades hace posible referirse a las mismas, lo cual es esencial para cualquier tipo de procesamiento simbólico.

Ingeniería de software: Designa el conjunto de técnicas destinadas a la producción de un programa de computadora, más allá de la sola actividad de programación. Forman parte de esta disciplina las ciencias computacionales y el manejo de proyectos, entre otros campos, propios de la rama más genérica denominada Ingeniería informática.

Ingeniería informática: Es la profesión que consiste en la aplicación de los fundamentos de la ciencia de la computación, la electrónica y la ingeniería de software, para el desarrollo de soluciones integrales de cómputo y comunicaciones, capaces de procesar información de manera automática.

Ingeniería inversa: Obtiene información a partir de un producto accesible al público, con el fin de determinar de qué está hecho, qué lo hace funcionar y cómo fue fabricado. Los productos más comunes que son sometidos a la ingeniería inversa son los programas de computadoras y los componentes electrónicos, pero básicamente casi cualquier proceso puede ser sometido a un análisis de Ingeniería Inversa.

Glosario de Términos

Internet: Es un conjunto descentralizado de redes de comunicación interconectadas, que utilizan la familia de protocolos TCP/IP, garantizando que las redes físicas heterogéneas que la componen funcionen como una red lógica única, de alcance mundial.

IP (**El Protocolo de Internet, en inglés Internet Protocol):** Es un protocolo no orientado a conexión usado tanto por el origen como por el destino para la comunicación de datos a través de una red de paquetes conmutados.

Lenguaje de marcado o lenguaje de marcas: Es una forma de codificar un documento que, junto con el texto, incorpora etiquetas o marcas que contienen información adicional acerca de la estructura del texto o su presentación.

Lenguaje de programación: Es un lenguaje que puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. Consiste en un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones.

Lenguajes declarativos: En ciencias computacionales, son aquellos lenguajes de programación en los cuales se le indica a la computadora que es lo que se desea obtener o que es lo que se está buscando.

Llave: Identificador usado en registros de tablas dentro de una base de datos. Cuando identifica en forma única a un registro es llamada llave primaria.

Metalenguaje: En lógica y lingüística, un metalenguaje es un lenguaje usado para hacer referencia a otros lenguajes.

Multimedia: Es un término que se aplica a cualquier objeto que usa simultáneamente diferentes formas de contenido informativo como texto, sonido, imágenes, animación y video para informar o entretener al usuario. También se puede calificar como medios electrónicos u otros medios que permiten almacenar y presentar contenido multimedia.

Navegador web (del inglés, navigator o browser): Es una aplicación software que permite al usuario recuperar y visualizar documentos de hipertexto, comúnmente descritos en HTML, desde servidores web de todo el mundo a través de Internet.

Nodo: En informática, un nodo es "Punto de intersección o unión de varios elementos que confluyen en el mismo lugar". Ejemplo: En una red de ordenadores cada una de las máquinas es un nodo, y si la red es Internet, cada servidor constituye también un nodo.

Página web: Es una fuente de información adaptada para la WWW y accesible mediante un navegador de Internet. Esta información se presenta generalmente en formato HTML y puede contener hiperenlaces a otras páginas web.

Glosario de Términos

Paquete: Es un grupo de información que consta de dos partes: los datos propiamente dichos y la información de control, en la que está especificado la ruta a seguir a lo largo de la red hasta el destino del paquete.

Paradigma de programación: Representa un enfoque particular o filosofía para la construcción del software. No es mejor uno que otro sino que cada uno tiene ventajas y desventajas. También hay situaciones donde un paradigma resulta más apropiado que otro.

POO: Es un paradigma de programación que usa objetos y sus interacciones, para diseñar aplicaciones y programas de computadora.

Procedimiento almacenado (**stored procedure en inglés**): Es un programa o procedimiento el cual es almacenado físicamente en una base de datos. Generalmente son escritos en un lenguaje de bases de datos propietario como PL/SQL para Oracle database o PL/PgSQL para PostgreSQL.

Programación funcional: Es un paradigma de programación declarativa basado en la utilización de funciones matemáticas.

Protocolo de red o Protocolo de Comunicación: Es el conjunto de reglas que especifican el intercambio de datos u órdenes durante la comunicación entre las entidades que forman parte de una red.

Red de computadoras, también llamada red de ordenadores o red informática: Es un conjunto de equipos (computadoras y/o dispositivos) conectados por medio de cables, señales, ondas o cualquier otro método de transporte de datos, que comparten información, recursos y servicios, etc.

Renderizado: Toma del contenido (HTML, XML, imágenes, etc.) y la información de formato (CSS, etc.) para posteriormente crearse una representación visual de una página web o archivo de datos.

Réplica: Es el proceso de duplicar o mas los datos existentes en una base de datos a otra(s). En muchos entornos se usan la réplica por espejo y la réplica por fragmentos de datos.

Servidor: En informática, un servidor es un tipo de software que realiza ciertas tareas en nombre de los usuarios. El término servidor ahora también se utiliza para referirse al ordenador físico en el cual funciona ese software, una máquina cuyo propósito es proveer datos de modo que otras máquinas puedan utilizar esos datos.

Sincronización: Se refiere al proceso de propagación de los cambios en los datos y el esquema entre la fuente de datos y los destinos después de haber aplicado la instantánea inicial en el destino.

Software: Conjunto de los componentes lógicos necesarios para hacer posible la realización de una tarea específica, en contraposición a los componentes físicos del sistema (hardware).

Glosario de Términos

Software: Se refiere al equipamiento lógico o soporte lógico de un computador digital, comprende el conjunto de los componentes lógicos necesarios para hacer posible la realización de una tarea específica.

SQL: Es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas. Una de sus características es el manejo del álgebra y el cálculo relacional permitiendo lanzar consultas con el fin de recuperar, de una forma sencilla, información de interés de una base de datos, así como también hacer cambios sobre la misma.

***TCP/IP o* familia de protocolos de Internet:** Es un conjunto de protocolos de red en la que se basa Internet y que permiten la transmisión de datos entre redes de computadoras. En ocasiones se le denomina "conjunto de protocolos TCP/IP", en referencia a los dos protocolos más importantes que la componen: Protocolo de Control de Transmisión (TCP) y Protocolo de Internet (IP).

TCP: Es un protocolo de comunicación orientado a conexión y fiable del nivel de transporte.

Transacción: Es un conjunto de órdenes que se ejecutan formando una unidad de trabajo, es decir, en forma indivisible o atómica. Cuando por alguna causa el sistema debe cancelar la transacción, empieza a deshacer las órdenes ejecutadas hasta dejar la base de datos en su estado inicial (llamado punto de integridad), como si la orden de la transacción nunca se hubiese realizado.

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

Tupla: En la teoría de bases de datos, una tupla se define como una función finita que mapea (asocia unívocamente) los nombres con algunos valores.

UNIX: Es un sistema operativo portable, multitarea y multiusuario; desarrollado en 1969, por un grupo de empleados de los laboratorios Bell de AT&T.

W3C (World Wide Web Consortium): Es un consorcio internacional que produce estándares para la WWW.

World Wide Web o Red Global Mundial (WWW): Es un sistema de documentos de hipertexto y/o hipermedios enlazados y accesibles a través de Internet. Con un navegador Web, un usuario visualiza páginas web que pueden contener texto, imágenes, vídeos u otros contenidos multimedia, y navega a través de ellas usando hiperenlaces.

XML: *Extensible Markup Language* (lenguaje de marcas extensible), es un metalenguaje extensible de etiquetas desarrollado por el W3C, que permite definir la gramática de lenguajes específicos.