Universidad de las Ciencias Informáticas

Facultad 6

Título: "Sistema de recopilación de datos para el análisis de las interacciones proteína – ligando**"**

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

Autores:

Yanet Marrero Vargas

Felipe Rodriguez Arias

Yunior Bauta Pentón

Tutores:

MsC**.** Longendri Aguilera Mendoza

Dr. Ernesto Moreno Frías

Ing. Reynaldo Alvarez Luna

Junio 2009

"La ciencia no es. . . ni misterio de los iniciados, ni privilegio de los aristócratas de la mente, sino el medio único que tiene el hombre de explicarse las leyes de la vida. . ."

José Martí.

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 2008.

Yanet Marrero Vargas **Felipe Rodriguez Árias**

Firma del Autor. Firma del Autor

Yunior Bauta Pentón

Firma del Autor.

Longendri Aguilera Mendoza **Ernesto Moreno Frías**

Firma del Tutor. Firma del Tutor

Reynaldo Alvarez Luna

Firma del Tutor

DATOS DE CONTACTO

MsC. Longendri Aguilera Mendoza
Universidad de las Ciencias Informáticas, Ciudad de la Habana, Cuba.
Email: loge@uci.cu

Dr. Ernesto Moreno Frías
Centro de Inmunología Molecular, Ciudad de la Habana, Cuba.
Email: emoreno@cim.sld.cu

Ing. Reynaldo Alvarez Luna
Universidad de las Ciencias Informáticas, Ciudad de la Habana, Cuba.
Email: rluna@uci.cu

Agradecimientos:

Ante todo a Fidel Castro Ruz, por darnos la posibilidad de estudiar en una universidad como esta y confiar siempre en nosotros como su tropa del futuro.

A nuestra familia y a nuestros amigos por siempre estar ahí cuando más los necesitamos.

A nuestro tutor y hermano mayor Loge por nunca decir que no, por siempre estar ahí como una fuente de conocimiento y por apoyarnos con su gran capacidad profesional y sus fabulosas ideas que nos guiaron en todo este proceso de tesis.

A Reinaldo y a Moreno por dedicarnos su tiempo y su apoyo.

Al tribunal y al oponente porque sus críticas, para nosotros constructivas, nos ayudaron a elaborar la tesis con mejor calidad.

A todos los compañeros y amigos que nos ayudaron de una forma u otra y a los que no. Muchas Gracias.

Dedicatoria Yanet:

A mami Darli por ser la persona más importante de mi vida, por darme todo su amor, por su entrega y sacrificio. Sólo le pido a la vida merecerte y que me de salud para demostrarte cuanto te quiero.

A mi novio Ale por darme todo su apoyo y comprensión, por estar presente siempre que lo he necesitado y por hacerme tan feliz. Sin tu ayuda nada de esto hubiera sido posible.

A mi abuela Primitiva, por ser mi segunda madre, por darme tanto cariño y por toda su dedicación durante todos estos años.

A mis compañeros de tesis Felipe y Yunior por ser personas tan maravillosas, por su optimismo y por demostrarme que siempre se puede. Gracias por soportarme todos estos meses.

Al Loge que más que mi tutor ha sido mi guía desde que estoy en la UCI. Gracias por ayudarme y por ser tan paciente conmigo.

A Dennys que a pesar de ser profesor me ha demostrado que es mi amigo incondicionalmente. Le agradezco por haberme ayudado tanto.

A mi tío Rupertico, a mi primita Evelyn y a mi abuelo que siempre están pendientes de mí.

A mis suegros que de una forma u otra ayudaron a que esto fuera posible.

A mis amistades de la UCI por darme tantos momentos de alegría, especialmente a Danay y a Yitsy que me ayudaron mucho con la tesis.

A todos los que hicieron posible que este sueño se hiciera realidad y a los que no también.

Dedicatoria Felipe:

A mis queridos padres que son la fuerza que me inspira la voluntad de ser mejor cada día, el consejo a tiempo, la reflexión exacta y oportuna, gracias por darme la posibilidad de vivir.

A Yunior por ser tan testarudo, tan trabajador, y por creer que siempre se podía, por su fortaleza y por saber reponerse de los malos momentos.

A Yanet por soportarme, por su dedicación y sus resultados; por ser la Juana de Arco de este equipo.

Al Loge por saber guiarnos por el camino correcto y enseñarnos que siempre las cosas se pueden hacer mejor.

Al doctor Moreno por sus enseñanzas y por dedicarnos su preciado tiempo.

A Reinaldo por su ayuda desinteresada.

A mi hermano por regalarme sus historias y su sonrisa, orgulloso de estar a tu lado.

A mi novia Lisbeth por ser mi mano derecha en cada momento, por la dedicación que inspiró en mí, por el apoyo desmedido.

A toda mi familia, en especial a todos mis abuelos, por la sobrada naturalidad y fuerza en el de cursar de sus vidas.

A todos mis amigos.

A la gente que me quiere y la que no.

A los queridos compatriotas de mi tierra, que han demostrado ser ejemplo de amigos, donde quiera que esté los llevaré conmigo.

A la gente de mi antiguo grupo 5, por su sencillez.

A todos mis profesores de la carrera que me regalaron su granito de arena para que hoy llegara a ser quien soy.

Dedicatoria Yunior:

A mami y a papi que son mi vida y que debo a ellos todo lo que soy tanto profesional como en lo personal, nunca los voy a defraudar porque son mi inspiración y mi ejemplo a seguir en todo momento.

A mi hermano Pedrito por ser mi fuerte, mi apoyo y mi confianza.

A mis hermanas Amidelis y Beatriz por apoyarme y confiar en mí.

A mi abuela Lidia y mi abuelo Rubén que siempre me han apoyado en todo.

A mi abuelita Laudelina la viejita más linda y buena de todo Santo Domingo.

A mi esposa por ser especial para mí, y a mi nueva familia de La Habana: Julita, Rafa, Neida, Lidia, Rubén, Yusniel por aceptarme y apoyarme tanto.

Y a mis compañeros de tesis Yanet y Felipe que con su dedicación, entrega y sacrificio han logrado cumplir este gran sueño.

A mi tutor y amigo Loge a quien le deseo lo mejor del mundo porque en realidad se lo merece.

Al Doctor Ernesto Moreno por todo el tiempo que nos dedicó.

A Reinaldo por su apoyo incondicional.

Resumen

Los investigadores del Centro de Inmunología Molecular (CIM) realizaron el estudio de las interacciones proteína - ligando a través de una compilación de archivos extraídos del Protein Data Bank (PDB) para obtener información que aparece implícita en los mismos. Como resultado implementaron un algoritmo que se puede utilizar tanto para la extracción de conocimientos, como para poner a prueba distintos algoritmos de acoplamiento. También se diseñó una base de datos para almacenar los resultados de las transformaciones hechas a los ficheros PDB.

El centro no cuenta con un sistema capaz de obtener los datos provenientes de la fuente PDB, transformarlos y cargarlos a una base de datos relacional para que puedan ser consultados. Además la base de datos diseñada no cumple los requerimientos actuales. En este trabajo se presenta el sistema de recopilación de datos para el análisis de las interacciones proteína - ligando implementado como solución para el CIM. Este sistema consiste en una aplicación escritorio que permite extraer, transformar y almacenar en una base de datos los archivos cuyo formato sea PDB, y una aplicación Web para consultar el resultado.

Palabras claves: Protein Data Bank, base de datos, interacciones proteína-ligando.

Índice

Índice de figuras

Introducción

Los medicamentos han sido utilizados por el hombre desde la prehistoria. La primera relación de fármacos con instrucciones para su elaboración, o farmacopea, apareció en la actual ciudad alemana de Nuremberg en 1546. Se define un fármaco o medicamento como un producto químico que se emplea en el diagnóstico, prevención o tratamiento de enfermedades. [1]

En el diseño de nuevos fármacos la primera etapa es la elección del objeto de estudio, también conocido como molécula blanco, sobre la que actuará el medicamento para lograr el efecto deseado. Las diversas investigaciones sobre fármacos han revelado que la acción terapéutica de estos incide principalmente en las proteínas, por lo que son la primera opción para fungir como molécula blanco. Una vez preparada la molécula blanco cuya función se desea alterar, es necesario realizar una búsqueda de las moléculas orgánicas más afines que logren actuar como ligandos potentes.

Existe un método de simulación por computadora para el desarrollo de fármacos con resultados exitosos: Método de reconocimiento molecular *in sílico* o Docking. Este es ampliamente utilizado en el mundo por compañías privadas e instituciones de investigación pública para el hallazgo de nuevos compuestos con efectos terapéuticos [2]. El método permite, además de las construcciones, la evaluación de los complejos moleculares resultantes.

Las investigaciones y las técnicas en este campo van en ascenso, por lo que cada día aparecen complejos proteína - ligando nuevos y las bases de datos donde se guarda esta información se enriquecen considerablemente. Los grupos de investigación de esta temática en el mundo, almacenan sus resultados en el Protein Data Bank (fuente principal de estructuras tridimensionales de proteínas), determinadas por métodos de difracción de rayos X o por resonancia magnética nuclear. Actualmente este banco de datos contiene información de más de cincuenta mil estructuras moleculares.

Recientemente, varias bases de datos se han hecho públicas para seguir la comprensión de las interacciones proteína - ligando, utilizando el banco de datos Protein Data Bank (PDB) como apoyo; no obstante, es difícil de probar los métodos de acoplamiento sobre una gran variedad de los complejos. De ahí, que lograr una base de datos la cual reúna toda la información respecto a los complejos moleculares proteína-ligando permitirá acelerar los procesos de selección de nuevos fármacos.

En la década del 80, comienzan a crearse en Cuba centros biotecnológicos entre los que se destacan el Centro de Investigaciones Biológicas (CIB), Centro de Ingeniería Genética y Biotecnología (CIGB), Centro de Inmunoensayo y el Centro de Inmunología Molecular (CIM). Este último es un centro donde opera una instalación de propósitos múltiples para la investigación y fabricación de productos terapéuticos. Por lo que investigadores del centro realizaron la compilación de un gran conjunto de

complejos de proteína – ligando extraídos del PDB, los cuales fueron adaptados, tanto para la extracción de conocimientos, como para poner a prueba distintos algoritmos de acoplamiento.
Se diseñó una base de datos para almacenar los resultados de las transformaciones de dichos complejos, pero actualmente no satisface la necesidad para la que fue creada [3]. No se cuenta con un sistema para extraer los complejos proteína - ligando de la fuente PDB y realizarles las transformaciones para almacenarlos en la base de datos de forma automática. Por otra parte, tampoco se cuenta con una aplicación informática para que estos datos sean consultados por los investigadores interesados.
Por lo anteriormente expuesto se plantea como **problema científico**: ¿Cómo contribuir al proceso de recopilación de datos de la fuente *Protein Data Bank* para almacenarlos en una base de datos y que puedan ser consultados posteriormente?

Por lo que el **objeto de estudio** es: La fuente de datos PDB.

El **campo de acción**: El proceso de extracción, transformación y carga de datos provenientes del PDB.

Necesidad de desarrollar un sistema para la extracción de datos provenientes del PDB y les realice las transformaciones para brindar información, que no está explícita en la fuente original, y permita almacenarlos en una base de datos para que sean consultados por usuarios.

El **objetivo general**: Desarrollar un sistema para extraer datos provenientes del PDB y les realice las transformaciones para almacenarlos en una base de datos que pueda ser consultada posteriormente.

Para dar cumplimiento al objetivo general se derivan **los objetivos específicos** siguientes:

- ✓ Realizar el análisis del sistema.
- ✓ Diseñar el sistema.
- ✓ Rediseñar la base de datos del CIM.
- ✓ Implementar el sistema.

Para alcanzar el cumplimiento de los objetivos específicos se trazaron las siguientes **tareas** de la investigación:

- ✓ Familiarización con el formato de archivos de la fuente Protein Data Bank.

- ✓ Familiarización con las principales tecnologías y mecanismos de extracción, transformación, carga y limpieza de datos.
- ✓ Familiarización con el estudio realizado por los investigadores del CIM.
- ✓ Estudio y selección de las metodologías y herramientas para el desarrollo del sistema.
- ✓ Descripción de los requerimientos.
- ✓ Selección de los patrones de diseño.
- ✓ Diseño de clases.
- ✓ Diseño de diagramas de interacción.
- ✓ Rediseño de la base de datos que cumpla con los requerimientos actuales del CIM.
- ✓ Construcción de los diagramas de componentes del sistema.
- ✓ Implementación de los componentes del sistema.

Aportes prácticos esperados del trabajo.

El sistema será introducido en el Centro de Inmunología Molecular, con lo cual suplirá la falta de un sistema automático para la extracción, transformación y carga de los datos provenientes del PDB en una base de datos relacional. Además de facilitar la consulta de los datos, permitirá el posterior análisis de los mismos.

Estructuración del contenido con una breve explicación de sus partes.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

En este capítulo se hará una familiarización con el formato de archivos de la fuente PDB. Se analizarán los mecanismos de extracción, transformación, carga y limpieza de datos. Se tratarán las variadas tendencias y tecnologías que se utilizarán para el desarrollo del sistema y el rediseño de la base de datos. Se caracterizarán los posibles patrones de arquitectura y los patrones de diseño a utilizar en el sistema.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

En este capítulo se describe el sistema propuesto para la situación problemática. Se presentan las características y funcionalidades del sistema a partir de los requisitos funcionales y no funcionales. Se

identifican los actores que interactúan con el sistema y se presentan los diagramas de casos de uso del sistema con sus descripciones.

CAPÍTULO 3: DISEÑO DEL SISTEMA

En este capítulo se mostrarán los artefactos de diseño generados durante el flujo de trabajo de Análisis y Diseño para el sistema. Se describirá en términos del lenguaje de los programadores, cada una de las clases, con atributos y métodos, y la relación entre ellas. Esta descripción se encuentra detallada en las clases del diseño y los diagramas de secuencia generados para la aplicación de escritorio y la aplicación Web. Se realizará además el rediseño de la base de datos y se mostrará la vista lógica de la misma que cumple con los requerimientos actuales del CIM. Se presentará el modelo de despliegue para mostrar la distribución física del sistema.

CAPÍTULO 4: IMPLEMENTACIÓN DEL SISTEMA

En este capítulo se hará una descripción de cómo se implementarán los elementos del Modelo de Diseño en termino de componentes. Se mostrará la vista lógica de implementación del sistema y los diagramas de componente generados para cada caso de uso. Se describirán los algoritmos más importantes del sistema y se mostrarán algunas imágenes del mismo.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

INTRODUCCIÓN

En este capítulo se hará una familiarización con el formato de de la fuente PDB. Se analizarán los mecanismos de extracción, transformación, carga y limpieza de datos. Se tratarán las variadas tendencias y tecnologías que se utilizarán para el desarrollo del sistema y el rediseño de la base de datos. Se caracterizarán los posibles patrones de arquitectura y los patrones de diseño a utilizar en el sistema.

1.1 PROTEIN DATA BANK

El Protein Data Bank es la fuente de información de las estructuras en tercera dimensión de proteínas más grande del mundo. Se estableció en 1971 y contenía solamente siete estructuras. Este banco de datos se ha ido enriqueciendo desde entonces y en la actualidad comprende más de cincuenta mil estructuras moleculares. Aumenta el número de complejos proteína - ligando a medida que investigadores en el mundo obtienen nuevos descubrimientos y los almacenan en este banco de datos. Tal es el caso que semanalmente son liberadas nuevas estructuras.

El Protein Data Bank brinda la posibilidad de descargar las estructuras biológicas que contiene en varios formatos de archivos. Los mismos pueden ser: PDB, mmCIF, PDBML/XML, FASTA, entre otros, aunque los anteriores son los más utilizados.

A continuación se describen brevemente los mismos:

- ✓ El formato de archivo PDB muestra casi la totalidad de la estructura de la proteína.
- ✓ El formato de archivo PDBML/XML es una alternativa para archivos PDB, brinda la representación de la estructura de archivos macromoleculares de datos en XML.
- ✓ El formato de archivo mmCIF es archivo de información cristalográfico, está diseñado para hacer frente a las limitaciones del formato PDB.
- ✓ El formato Fasta es una alternativa para archivos PDB, brinda la representación de la estructura de archivos macromoleculares de datos en texto.

El formato de archivo PDB es el más conocido y utilizado en el mundo. En la investigación realizada por investigadores del CIM, PDB fue el formato seleccionado para estudiar las interacciones proteína-ligando; por tal motivo, en el sistema a desarrollar los datos a recopilar serán los contenidos en ese formato.

1.2 PROCESO DE EXTRACCIÓN, TRANSFORMACIÓN Y CARGA DE DATOS

En el estudio de los procesos de recopilación de gran variedad de datos digitalizados desde múltiples fuentes para cargarlos en una fuente destino juegan un papel importante los mecanismos de extracción, transformación, carga y limpieza de datos. Un proceso ETL (Extract – Transform - Load) organiza el flujo de datos entre diferentes fuentes o sistemas y aporta los métodos y herramientas necesarias para mover datos hacia un formato único o almacén de datos para su posterior análisis.

Entre las más populares herramientas ETL del mercado pueden ser citadas algunas como: Clover ETL, Jasper ETL, Pentaho Data Integration (Kettle ETL), entre muchas otras.

El proceso de *extracción*: La primera parte del proceso ETL consiste en extraer los datos desde los sistemas de origen. La extracción convierte los datos a un formato preparado para iniciar el proceso de transformación.

Una parte intrínseca del proceso de extracción es la de analizar los datos extraídos, de lo que resulta un chequeo que verifica si los datos cumplen la pauta o estructura que se esperaba. De no ser así los datos son rechazados.

El proceso de *transformación*: Se aplican una serie de reglas de negocio o funciones sobre los datos extraídos para convertirlos en datos que serán cargados. Algunas fuentes de datos requerirán alguna pequeña manipulación como la aplicación de cualquier forma, simple o compleja, de validación de datos, y la consiguiente aplicación de la acción que en cada caso se requiera.

La *carga* de los datos en la fuente destino: Este es el momento en el cual los datos de la fase anterior (*transformación*) son cargados en el sistema destino. Este proceso abarca una amplia variedad de acciones diferentes tales como mantener un historial de los registros de manera que se pueda hacer una auditoría de los mismos y disponer de un rastro de toda la historia de un valor a lo largo del tiempo almacenados en *logs*.

Se determinó utilizar el mecanismo de funcionamiento de los procesos ETL y no una herramienta ETL para el desarrollo del sistema debido a que las transformaciones a realizarse sobre los archivos PDB para la obtención de información biológica implícita son muy complejas.

1.3 ESTUDIO REALIZADO POR INVESTIGADORES DEL CIM

Los investigadores del CIM realizaron la compilación de un gran conjunto de complejos proteína – ligando extraídos del PDB, los cuales fueron adaptados, tanto para la extracción de conocimientos en

búsquedas a gran escala, como para poner a prueba distintos algoritmos de acoplamiento. El término "a gran escala", es empleado para referirse al uso de miles de proteínas que son complejos ligando identificados en el PDB y procesados de modo automático.

Las alteraciones realizadas a los complejos proteína-ligando fueron resultado del algoritmo complex_info implementado en Fortran, debido a que este lenguaje de programación presenta excelente soporte para la utilización de funciones matemáticas de cálculo. Cuando un fichero PDB es procesado el resultado será una copia en disco duro de ficheros ligand.pdb, contact.table y .mor por cada ligando.

El algoritmo permite obtener información implícita de los archivos PDB, por ejemplo, es capaz de obtener información detallada de los sitios activos de la molécula en cuestión, por donde se producen los enlaces con otras sustancias. Además, cuando identifica un ligando es capaz de determinar el nivel de compenetración que existe en el complejo, los átomos de cada estructura que intervienen en dicha unión y la distancia a la que se encuentran, entre otras características.

Los datos obtenidos de la fuente PDB deberán ser transformados por el algoritmo implementado por los investigadores del CIM y posteriormente serán cargados en la base de datos.

1.4 JMOL

El desarrollo de la bioinformática facilitó la creación de herramientas capaces de visualizar estructuras moleculares en tercera dimensión. Así surgen varios visualizadores moleculares como RasMol, Chime, PyMol y Jmol.

Jmol es un visor de moléculas gratuito y de código abierto para estudiantes, profesores e investigadores en química y bioquímica. Es multiplataforma, compatible con sistemas Windows, Mac OS X y Linux/Unix [4].

Jmol es una aplicación desarrollada en Java que permite obtener una visión tridimensional de una estructura química. Los átomos que componen los compuestos pueden observarse prácticamente desde cualquier ángulo. Jmol se compone principalmente de una herramienta de escritorio y de un complemento para el navegador Web (JmolApplet) que permite explorar las estructuras moleculares alojadas en cualquier página. Jmol incluye además su propia utilidad para generar una página Web a partir de un archivo compatible con el programa y permite añadir algunos detalles a la vista 3D de las estructuras, entre ellos: mediciones entre átomos o el símbolo, el número y el nombre del elemento químico al que representan. [5]

La visualización de los complejos proteína - ligando en el sistema podrá hacerse a través de JmolApplet, por las características expuestas anteriormente, por su capacidad de integrarse en aplicaciones Web y por ser este el visor de estructuras moleculares utilizado por el CIM.

1.5 METODOLOGÍA DE DESARROLLO DE SOFTWARE

Una metodología de desarrollo de software es un conjunto de pasos y procedimientos que deben seguirse para desarrollar software [6]. Entre las metodologías de desarrollo de software se encuentran el Proceso Unificado de Desarrollo (RUP) y el Proceso Unificado Abierto (OpenUP).

1.5.1 Proceso Unificado de Desarrollo (RUP)

La metodología RUP, llamada así por sus siglas en inglés Rational Unified Proccess, es el proceso unificado de desarrollo de software que plantea quién hace qué, cuándo y cómo. Se actualiza constantemente para tener en cuenta las mejores prácticas y utiliza además UML como lenguaje de modelado.

Tiene como características principales: estar dirigido a casos de usos, centrado en la arquitectura y es iterativo e incremental. Una característica importante es que permite corregir errores en cada iteración y es flexible a cambios en los requerimientos.

1.5.2 Proceso Unificado Abierto (OpenUP)

Es una versión simplificada del IBM Rational Unified Process (RUP) optimizado para proyectos pequeños. Conserva las características esenciales de RUP, que incluye el desarrollo iterativo, casos de uso y escenarios de conducción de desarrollo, gestión de riesgos, y el enfoque centrado en la arquitectura. Ya que es apropiado para proyectos pequeños y de bajos recursos permite disminuir las probabilidades de fracaso en los proyectos pequeños e incrementar las probabilidades de éxito. Permite detectar errores tempranos a través de un ciclo iterativo. Evita la elaboración de documentación, diagramas e iteraciones innecesarios requeridos en la metodología RUP. Por ser una metodología ágil tiene un enfoque centrado al cliente y con iteraciones cortas. [7]

Se selecciona OpenUp como la metodología a utilizar para el desarrollo del sistema, por ajustarse las características presentadas anteriormente a las del presente trabajo y por ser esta la metodología de desarrollo de software utilizada por el Grupo de Bioinformática de la Universidad de las Ciencias Informáticas.

1.6 LENGUAJE DE MODELADO

El Lenguaje Unificado de Modelado (UML) es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. El lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un mecanismo estándar. Está pensado para ser utilizado en herramientas interactivas de modelado visual que tengan generadores de código así como generadores de informes. Pretende dar apoyo a la mayoría de los procesos de desarrollo orientados a objetos.

1.7 HERRAMIENTA PARA EL MODELADO

El modelado de sistemas software es una técnica para tratar con la complejidad inherente a estos sistemas. El uso de modelos ayuda al ingeniero de software a especificar, visualizar, construir y documentar los artefactos del sistema a construir. Además, los sistemas de nivel de abstracción mayor pueden utilizarse para la comunicación con el cliente. Las herramientas de modelado pueden ayudar a verificar la corrección del modelo. Dentro de las herramientas más usadas se encuentran el Rational Rose y el Visual Paradigm.

1.7.1 Rational Rose

Es la herramienta CASE desarrollada por los creadores de UML Booch, Rumbaugh y Jacobson, provee un modelado basado en UML y cubre todo el ciclo de vida de un proyecto: concepción y formalización del modelo, construcción de los componentes, transición de los usuarios y certificación de las distintas fases y entregables.

1.7.2 Visual Paradigm

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación [8]. Genera código para un gran número de lenguajes de programación entre los que se encuentra Java y permite integración con varias

herramientas de Java. Además, es multiplataforma. Se selecciona Visual Paradigm como herramienta para el modelado por las características que presenta y por ser la utilizada por el Grupo de Bioinformática de la Universidad de las Ciencias Informáticas.

1.8 TECNOLOGÍAS Y HERRAMIENTAS PARA EL DESARROLLO

1.8.1 LENGUAJES DE PROGRAMACIÓN

1.8.1.1 Java

El principal objetivo de los diseñadores de Java, y dado el gran crecimiento de las redes en los últimos años, fue desarrollar un lenguaje cuyas aplicaciones una vez compiladas pudiesen ser inmediatamente ejecutables en cualquier máquina y sobre cualquier sistema operativo. [9]

Los diseñadores de Java trataron de mantener las facilidades básicas del lenguaje en un mínimo y proporcionar un gran número de extras con las librerías de clases.

Es un lenguaje de programación seguro, que no puede acceder a los recursos del sistema de manera incontrolada. Por este motivo se eliminó la posibilidad de manipular la memoria mediante el uso de punteros y la capacidad de transformación de números en direcciones de memoria.

Java permite la ejecución de rutinas externas que pueden estar escritas en cualquier otro lenguaje, por ejemplo FORTRAN, con el método Process exec(String arg) de la clase Runtime.

1.8.1.2 JSP

Como parte de la familia de la tecnología Java, JSP permite un desarrollo rápido de aplicaciones basadas en Web que son independientes de la plataforma. JSP separa la interfaz de usuario de la generación de contenidos, permitiendo a los diseñadores cambiar el diseño de páginas sin alterar la dinámica subyacente del contenido.

1.8.2 FRAMEWORK

Los framework son programas, componentes, utilitarios (código en general) que facilitan el desarrollo de aplicaciones de software relativamente estandarizadas (o partes de estas) proporcionando el diseño y a veces componentes auxiliares (librerías o toolboxes.) [10]. Son soluciones que implementan patrones y ayudan a desarrollar funciones dentro de las aplicaciones a un nivel

superior. Tienen la funcionalidad de abstraer complejas implementaciones y hacer el desarrollo más ágil, ya que proveen soluciones a problemas comunes. Son como un conjunto de librerías que usan las aplicaciones. Constituyen elementos a tener en cuenta a la hora de modelar una arquitectura pues hacen que se cubran en gran medida los objetivos con los que debe cumplir la misma.

1.8.2.1 Spring

Spring Framework es una librería open source que ofrece facilidades avanzadas para la organización de servicios en una aplicación Java [11]. Es considerado el primer framework basado en soluciones que requieren inyección de dependencias. Recientemente (a partir del Spring 2.0) ha logrado separar (fuera de la lógica de negocio) parte de la arquitectura, mediante el uso de anotaciones de documentos escritos en XML, lo que constituye una fortaleza en el diseño de componentes de software basados en servicios. Tiene además sus propios frameworks para el manejo de los aspectos esenciales a tener en cuenta en el desarrollo:

- ✓ Acegi: para la seguridad.
- ✓ Spring MVC: para el desarrollo Web.
- ✓ Spring-AOP: para el desarrollo de Aspect Oriented Programming.
- ✓ Spring-WebServices: para el uso de servicios Web.
- ✓ Spring: frameworks de inyección de dependencias.
- ✓ Spring-JDBC: framework de persistencia.
- ✓ Spring Web Flows: framework para definir Web flows.

Spring Framework es recomendable para el desarrollo de aplicaciones, pues constituye un contenedor ligero que gestiona los objetos de las mismas, manteniendo el código limpio y claro. Los objetos gestionados no necesitan implementar o extender funcionalidades especiales, por cuanto es un mecanismo totalmente transparente. Además, los servicios que brinda Spring son fundamentales en cualquier aplicación y abstraen al programador de codificar cientos, miles de líneas, esto hace que aumente la productividad en los proyectos actuales que lo utilicen.

1.8.3 JAVA WEB START

Java Web Start permite descargar y ejecutar aplicaciones de escritorio desde la Web.

- ✓ Permite activar las aplicaciones escritorio con un simple clic.
- ✓ Garantiza que se está ejecutando la última versión de la aplicación escritorio.
- ✓ Elimina complejos procedimientos de instalación o actualización.

El software de Java Web Start es un componente del entorno de ejecución de Java (JRE) y se instala con el JRE. Cuando se descarga por primera vez una aplicación que utiliza la tecnología Java Web Start, el software de Java Web Start se ejecuta automáticamente y guarda la aplicación localmente, en la memoria caché del equipo. De este modo, las subsiguientes ejecuciones son prácticamente instantáneas, ya que los recursos necesarios están disponibles de forma local. Cada vez que se inicia la aplicación, el software de Java Web Start comprueba si en la sede Web de la aplicación hay una versión nueva disponible; si es así, la descarga y la ejecuta de forma automática. [12]

1.8.4 HIBERNATE

Hibernate es una herramienta para la plataforma Java que facilita el mapeo de atributos entre una base de datos relacional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) que permiten establecer estas relaciones. [13]

Hibernate en su funcionamiento genera sentencias HQL y libera al desarrollador del manejo manual de los datos que resultan de la ejecución de dichas sentencias, manteniendo la portabilidad entre todas las bases de datos con un ligero incremento en el tiempo de ejecución.

Hibernate se distribuye como una herramienta de software libre, distribuida bajo los términos de la licencia GNU LGPL. Hibernate parte de una filosofía de mapear objetos Java, también conocidos como “POJOs” (Plain Old Java Objects). Una de las principales características de Hibernate es su flexibilidad, envolviéndolo todo bajo un marco de trabajo común. [14]

1.8.5 HERRAMIENTAS DE DESARROLLO

1.8.5.1 Eclipse

Eclipse es una poderosa herramienta que permite integrar diferentes aplicaciones para construir un entorno integrado de desarrollo (IDE) [15]. Es un potente entorno de desarrollo de Java. Usa java como lenguaje de programación ya que soporta la programación orientada a objetos (POO) y la implementación de aplicaciones resulta mucho más sencilla. Mediante Eclipse se puede crear diversas aplicaciones como son sitios Web, programas Java y Enterprise Java Beans. Su principal aplicación es JDT (Java Development Tool), herramienta para crear aplicaciones en Java. Otras aplicaciones pueden ser integradas a eclipse en forma de plugins, que son reconocidos automáticamente por Eclipse al iniciar el mismo.

Beneficios de la herramienta Eclipse:

- ✓ Es una herramienta de código abierto.
- ✓ Soporta la construcción de una variedad de herramientas para el desarrollo de aplicaciones.
- ✓ Soporta plugin para spring framework.
- ✓ Corre en una gran cantidad de sistemas operativos incluyendo Windows y Linux.

1.8.5.2 Apache Tomcat

Tomcat es un servidor Web con soporte de servlets JSPs. Incluye el compilador Jasper, que compila JSPs convirtiéndolas en servlets. El motor de servlets de Tomcat a menudo se presenta en combinación con el servidor Web Apache.
Tomcat puede funcionar como servidor Web por sí mismo, es usado como servidor Web en entornos con alto nivel de tráfico y alta disponibilidad. Dado que Tomcat fue escrito en Java, funciona en cualquier sistema operativo que disponga de la maquina virtual Java.

El lenguaje de programación seleccionado para el desarrollo del sistema fue Java y JSP. Se decidió utilizar el framewok spring y entre sus propios frameworks: Spring MVC, Spring-JDBC y los frameworks de inyección de dependencias así como su integración con hibernate para la capa de acceso a datos. El IDE de desarrollo escogido es eclipse y Apache Tomcat como servidor Web. Además se utilizará la tecnología Java Web Start por las bondades que ofrece.

1.10 GESTOR DE BASE DE DATOS POSTGRESQL

PostgreSQL es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (ORDBMS). PostgreSQL está ampliamente considerado como el sistema de bases de datos de código abierto más avanzado del mundo [16].
Características principales:

- ✓ Implementación del estándar SQL92/SQL99.
- ✓ Licencia BSD.
- ✓ Por su arquitectura de diseño, escala muy bien al aumentar el número de CPUs y la cantidad de RAM.
- ✓ Soporta transacciones y desde la versión 7.0, claves ajenas (con comprobaciones de integridad referencial).
- ✓ Tiene mejor soporte para triggers y procedimientos en el servidor.
- ✓ Incorpora una estructura de datos array.

- ✓ Incluye herencia entre tablas (aunque no entre objetos, ya que no existen), por lo que a este gestor de bases de datos se le incluye entre los gestores objeto-relacionales.
- ✓ Implementa el uso de rollback's, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz.
- ✓ Se pueden realizar varias operaciones al mismo tiempo sobre la misma tabla sin necesidad de bloquearla. [17]

Por las características presentadas anteriormente de PostgreSQL se selecciona como sistema gestor de base de datos para el sistema a desarrollar.

1.11 PATRONES DE ARQUITECTURA

1.11.1 MODELO-VISTA-CONTROLADOR

Spring introduce el patrón arquitectónico Modelo-Vista-Controlador (MVC) que divide una aplicación interactiva en 3 áreas: procesamiento, salida y entrada. Para esto, utiliza las siguientes abstracciones:
Modelo (Model): encapsula los datos y las funcionalidades. El modelo es independiente de cualquier representación de salida y/o comportamiento de entrada.
Vista (View): muestra la información al usuario. Pueden existir múltiples vistas del modelo. Cada vista tiene un componente controlador.
Controlador (Controller): reciben las entradas, usualmente como eventos que codifican los movimientos o pulsación de botones del ratón o pulsaciones de teclas. Los eventos son traducidos a solicitudes de servicio (“service requests”) para el modelo o la vista.
Se definen las clases dominio (Modelo) para que no tengan acoplamiento ni visibilidad directa respecto a las clases ventana (Vista) y para que los datos de la aplicación y de la funcionalidad se conserven en las clases de dominio, no en la ventana. Definen las clases manejadores (Controlador) para que procesen los eventos (peticiones) al sistema y redirecciones las clases dominio y ventana tanto el procesamiento como la visualización de resultados respectivamente.
Algunos de los principales beneficios son:

- ✓ Menor acoplamiento.
- ✓ Mayor cohesión.
- ✓ Las vistas proveen mayor flexibilidad y agilidad.
- ✓ Mayor facilidad para el desarrollo de clientes ricos en múltiples dispositivos y canales.
- ✓ Más claridad de diseño.
- ✓ Facilita el mantenimiento.

- ✓ Mayor escalabilidad. [16]

1.11.2 ARQUITECTURA EN CAPAS

La arquitectura en tres capas permite que el desarrollo de la aplicación se pueda llevar a cabo en varios niveles y en caso de algún cambio solo se ataca al nivel requerido sin tener que revisar entre código mezclado.

- ✓ *La capa de la Presentación*: Como su nombre lo indica, esta capa reúne todos los aspectos del software que tienen que ver con las interfaces y la interacción del usuario con las mismas.
- ✓ *La capa de negocio:* Es la capa que sirve de comunicación entre la capa de presentación y la capa de acceso a datos, es donde se implementa la lógica del negocio de la aplicación. Esta capa también recibe el nombre de la capa de la Lógica de la Aplicación.
- ✓ *La capa de acceso a datos:* Esta capa reúne todos los aspectos del software que tienen que ver con el manejo de los datos persistentes.

1.12 PATRONES DE DISEÑO

Los patrones de diseño son soluciones simples y elegantes a problemas específicos y comunes del diseño orientado a objetos. Son soluciones basadas en la experiencia y que se ha demostrado que funcionan.

1.12.1 Patrón Alta Cohesión

El patrón Alta Cohesión es una medida de cuan relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Una clase con baja cohesión hace muchas cosas no afines o un trabajo excesivo. La solución es asignar una responsabilidad de modo que la cohesión siga siendo alta.

1.12.2 Patrón Bajo Acoplamiento

El patrón Bajo Acoplamiento resuelve el problema de cómo dar soporte a una dependencia escasa y a un aumento de la reutilización. La solución es asignar una responsabilidad para mantener bajo acoplamiento.

1.12.3 Patrón Puente

El patrón Puente (Bridge): desacopla una abstracción de su implementación. Es un patrón GOF estructural utilizado para separar interfaces e implementación de las clases que prestan servicios y DAOs (Data Acces Object/clases de acceso a datos).

1.12.4 Patrón Factory

El patrón Factory es utilizado para la selección y retorno de una instancia de clase, de un número de clases similares basado en la información que se le indique a la clase Factory. Este patrón propone crear objetos por medio de distintos métodos, a partir de algunos atributos, pero sin especificar una clase concreta. Generalmente estos métodos son estáticos y reciben como parámetros, los valores a los atributos necesarios para crear los objetos.

1.12.5 Patrón Instancia Única

El patrón Instancia Única (Singleton) garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. Es un patrón GOF creacional utilizado para crear todas las instancias únicas de las clases controladoras, servicios, daos contenidas en el bean factory de Spring.

1.12.6 Patrón Facade

El patrón Facade trata de hacer una estructura fácilmente entendible que proporcione una forma de trabajar ordenada. Se basa en proporcionar al programador una clase sencilla con un grupo de métodos, uno para cada operación permitida y de modo que sean estos métodos los que internamente hagan las operaciones con el fin de llevar a cabo la correcta lógica de la aplicación. El patrón de diseño Facade sirve para proveer de una interfaz unificada sencilla que haga de intermediaria entre un cliente y una interfaz o grupo de interfaces más complejas.

1.12.7 Patrón DAO

El uso del patrón DAO ofrece varios beneficios para la persistencia de datos. Este patrón sirve para separar el acceso a datos de la lógica de negocio. Encapsula la fuente de datos. Oculta la API con la que se accede a los datos y centraliza todos los accesos a datos en una capa independiente.

1.12.8 Patrón Inyección de dependencias

Inyección de dependencias es una de las implementaciones del concepto Inversión de Control. Desde la perspectiva de Inversión de Control se reconoce que los objetivos involucrados en la lógica de negocio de una aplicación dependen de otros objetos de negocio, objetos de acceso a datos y recursos compartidos. Todos ellos se ejecutan en un ambiente denominado Contenedor Ligero, en cual es responsable de establecer las dependencias entres los objetos.

CONCLUSIONES

En el sistema a desarrollar los datos a recopilar de la fuente Protein Data Bank serán los contenidos en el formato de archivo PDB, se transformarán a través del algoritmo implementado por los investigadores del CIM y posteriormente serán cargados en la base de datos para ser consultados. Se determinó utilizar el mecanismo de funcionamiento de los procesos ETL en el desarrollo del sistema. La visualización de las estructuras moleculares se hará con JmolApplet. Se estableció la metodología OpenUp como la más apropiada y UML como lenguaje de modelado y Visual Paradigm como herramienta para el modelado. Se determinaron las tecnologías, herramientas y el sistema gestor de base de datos a utilizar en el presente trabajo.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

INTRODUCCIÓN

En este capítulo se describe el sistema propuesto para la situación problemática. Se presentan las características y funcionalidades del sistema a partir de los requisitos funcionales y no funcionales. Se identifican los actores que interactúan con el sistema y se presentan los diagramas de casos de uso del sistema con sus descripciones.

2.1 MODELACIÓN DEL SISTEMA

2.1.1 Solución informática propuesta

Considerando que el volumen de archivos PDB a recopilar que se encuentran en un servidor del CIM asciende a 6GB compactados, los que contienen treinta y seis mil ficheros al descompactarse; se propone realizar el proceso de extracción, transformación y carga de los mismos en la base de datos a través de una aplicación escritorio. De esta manera se garantiza que el procesamiento de los datos se realice en la PC cliente, lo que no se podría hacer con una aplicación Web. La base de datos podrá ser consultada a través de una aplicación Web. La integración de ambas aplicaciones estará garantizada con el uso de la tecnología Java Web Start.

2.1.2 Requisitos Funcionales y No Funcionales del Sistema

Requisitos Funcionales:

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir, y se mantienen invariables sin importar con que propiedades o cualidades se relacionen. Se muestran a continuación los requisitos funcionales del sistema:

R 1. Autenticar Usuario.
R 2. Registrar Investigador.
R 3. Visualizar Proteína.
R 4. Consultar Información de la Base de Datos.
  R 4.1. Realizar Consulta Predeterminada a la Base de Datos.
  R 4.2. Diseñar Nueva Consulta a la Base de Datos.

R 4.3. Consultar Vista Lógica de la Base de Datos.

R 4.4. Consultar Fichero PDB.

R 4.5. Consultar Información de Fichero PDB.

R 4.6. Consultar Información de Ligando.

R 4.7. Consultar Información de Sitio de Enlace.

R 5. Gestionar Consultas Predeterminadas.

R 5.1. Insertar Consulta Predeterminada.

R 5.2. Listar Consultas Predeterminadas.

R 5.3. Eliminar Consulta Predeterminada.

R 6. Exportar Resultado de Consulta.

R 6.1. Exportar Consulta en Formato .tar.gz.

R 6.2. Exportar Consulta en Formato ligand.pdb.

R 6.3. Exportar Consulta en Formato .contact.table.

R 7. Gestionar Usuarios.

R 7.1. Listar Usuarios.

R 7.2. Eliminar Usuario.

R 7.3. Crear Nuevo Administrador.

R 8. Gestionar Comentarios del Administrador.

R 8.1. Listar Comentarios.

R 8.2. Eliminar Comentario.

R.8.3. Responder Comentario.

R 9. Gestionar Comentarios del Investigador.

R 9.1. Redactar Nuevo Comentario.

R 9.2. Listar Comentarios Respondidos.

R 9.3. Eliminar Comentario Respondido.

R 9.4. Responder Comentario al Administrador.

R 10. Autenticar Administrador.

R 11. Actualizar la Base de Datos.

R 11.1. Almacenar un fichero PDB.

R 11.2. Almacenar directorio de ficheros PDB.

R 11.3. Almacenar ficheros PDB desde el servidor FTP.

Requisitos no funcionales:

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. A continuación se presentan los requisitos no funcionales identificados para el sistema.

- ✓ Requerimientos de software:

Para las PCs clientes que requieran la aplicación Web el Sistema Operativo Windows 95 o superior o Sistema Operativo Linux basados en distribuciones Debian. Los usuarios accederán al sistema utilizando uno de los siguientes navegadores: Mozilla 1.7 o superior, Firefox 0.9.3 o superior, Internet Explorer 7.0 o superior, Opera 7.23 o superior. Además se deberá contar con resoluciones de [1024x768pix – 800x600pix].
Para las PCs clientes que requieran la aplicación escritorio el Sistema Operativo Linux basados en distribuciones Debian. Máquina Virtual de Java versión 1.6 o superior. Librería libg2c0 para ejecutar aplicaciones desarrolladas en Fortran 77.
Para el Servidor de base de datos el Sistema Operativo Linux basados en distribuciones Debian. Postgres 8.3.
Para el Servidor Web el Sistema Operativo Linux basados en distribuciones Debian. Apache – Tomcat 5.5.
Para el Servidor Ftp: Sistema Operativo Linux basados en distribuciones Debian.

- ✓ Requerimientos de hardware:

Para las PCs clientes que requieran la aplicación escritorio se proponen los siguientes requisitos mínimos: 64Mb de RAM y 100Mb de espacio libre en el disco duro.
Los servidores como mínimo deberán contar con 256Mb de RAM y 40GB de disco duro.
Todas las máquinas implicadas deben estar conectadas a una red de al menos 100 Mbps de velocidad.

- ✓ Requerimientos de extensibilidad:

Se debe lograr un diseño adaptable al proceso ETL, con la capacidad de soportar reglas adicionales de extracción, transformación y carga de datos. Se debe brindar la posibilidad de modificar las funcionalidades existentes sin impactar en el resto de los requerimientos contemplados en el sistema.

- ✓ Requerimientos de restricciones en el diseño y la implementación:

Los nombres de clases deben ser simples y descriptivos, utilizando palabras completas y acrónimos o abreviaturas (a no ser que la abreviatura sea conocida, como DAOImpl o PDB) al igual que los métodos. Los nombres de variables deben ser significativos. La elección de un nombre de variable debe ser pensada para que un lector casual al verla comprenda su uso. El lenguaje de programación utilizado para la implementación será Java y se utilizará el entorno de programación brindado por el IDE Eclipse y el sistema gestor de base de datos será PostgreSQL.

- ✓ Requerimientos de apariencia o interfaz externa:

El sistema tendrá una apariencia amigable con colores de bajos tonos. Las páginas de la aplicación Web no se cargarán con mucha información, y contendrán solo las imágenes necesarias. Se hará uso de la simbología mediante íconos para indicar algunas funcionalidades del sistema.

- ✓ Requerimientos de usabilidad:

El sistema tendrá un ambiente sencillo y será fácil de manejar por los usuarios, incluso aquellos que no han tenido mucha experiencia en el trabajo con computadoras o con sistemas informáticos. Los usuarios finales de la aplicación deben tener un conocimiento previo acerca de los formatos de estructuras biológicas que el software maneja (formato PDB). Este nivel de conocimiento describe los límites apropiados para el uso del sistema.

- ✓ Requerimientos de soporte:

Se impartirá una preparación a los usuarios finales en la cual se explicará cómo se realizará el trabajo con el sistema.

- ✓ Requerimientos de portabilidad:

El sistema podrá admitir cambios en la base del sistema operativo, se usará sobre los sistemas operativos Windows de arquitectura 32bits o 64bits y cualquier distribución de GNU-Linux, exceptuando la aplicación escritorio que podrá ser ejecutado sólo en sistema operativo GNU-Linux.

- ✓ Requerimientos políticos-culturales

El idioma que se empleará en el sistema será el español, pero los reportes de las estructuras proteína-ligando se presentarán en ingles.

- ✓ Requerimientos de seguridad:

Las contraseñas deberán tener más de 5 caracteres de longitud. El sistema debe contar con una protección contra acciones no autorizadas o que puedan afectar la integridad de los datos. Los permisos para acceder al sistema serán dados en dependencia del rol del usuario.

2.1.3 Descripción de los actores del sistema

Actor	Descripción
Administrador	Es el encargado de mantener actualizada la base de datos y de elaborar las consultas comunes que un investigador puede hacer a la misma (consultas predeterminadas). Además de ser el responsable de responder las inquietudes o sugerencias que los investigadores puedan tener sobre el sistema.
Investigador	Es la persona que interactúa con el sistema para consultar información de la base de datos. Puede enviar mensajes sobre la aplicación, hacer sugerencias y plantear sus inquietudes a través de los mismos.

2.1.4 Diagramas de Casos de Uso del Sistema

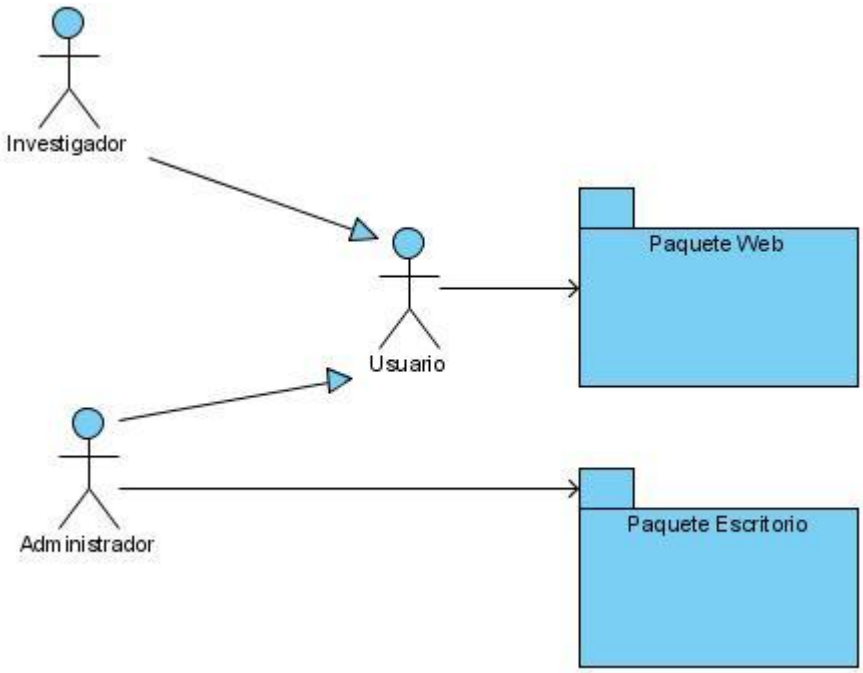


Fig. 1 Diagrama General de CUS

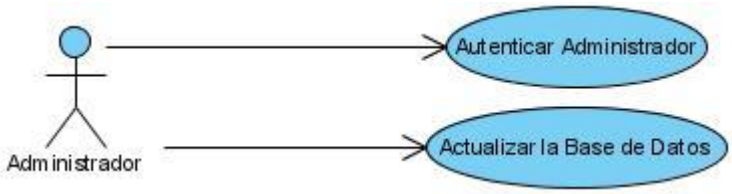


Fig. 2 Paquete Escritorio

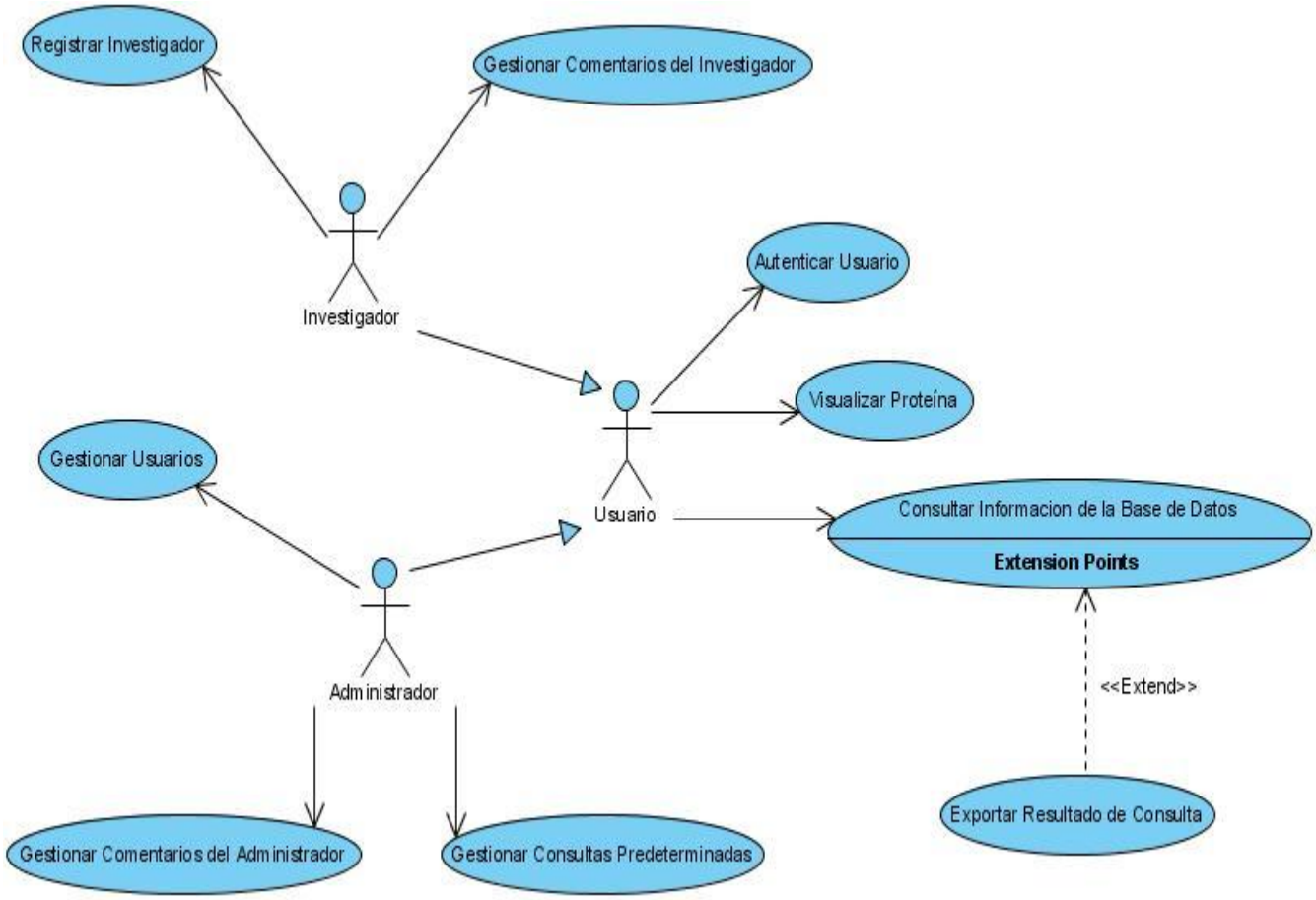


Fig. 3 Paquete Web

2.1.5 Descripción de los Casos de Uso del Sistema

Descripción del Caso de Uso: Consultar Información de la Base de Datos.

Nombre del CUS	Consultar Información de la Base de Datos.
Actores	Usuario (Inicia el CU).
Propósito	Realizar consultas a la Base de Datos.
Resumen	El caso de uso se inicia cuando el usuario selecciona la opción "CONSULTAS". El sistema muestra las interfaces correspondientes para dar cumplimiento al caso de uso. El usuario realiza las acciones necesarias. Finaliza el caso de uso.
Referencias	R 4.1, R 4.2, R 4.3, R 4.4, R 4.5, R 4.6, R 4.7.
Precondiciones	El usuario debe estar registrado en el sistema.
Curso Normal de los Eventos	

Sección: "General"	
Acciones del Actor	**Respuesta del Sistema**
1. El usuario del sistema se dirige al menú principal y selecciona la opción "CONSULTAS".	2. El sistema muestra un listado de consultas predeterminadas y la posibilidad de ir a la sección "Consultas Avanzadas".
Sección: "Realizar Consulta Predeterminada a la Base de Datos"	
Acciones del Actor	**Respuesta del Sistema**
3. El usuario selecciona una consulta del listado de consultas predeterminadas y la opción "Buscar".	4. El sistema verifica que la consulta sea correcta.
	5. El sistema muestra la información contenida en la base de datos relacionada con la consulta.
6. El usuario selecciona el fichero del cual desea ver la información que contiene. Ir al paso 9.	
Sección: "Diseñar Nueva Consulta a la Base de Datos"	
Acciones del Actor	**Respuesta del Sistema**
3. El usuario selecciona la opción "Consultas Avanzadas" para diseñar una nueva consulta.	4. El sistema muestra una interfaz brindando las posibilidades necesarias para que el usuario diseñe una nueva consulta (El sistema muestra las tablas de la base de datos con los respectivos atributos para facilitar el diseño de las consultas) y brinda la posibilidad de ver el diseño de clases de la base de datos.
5. El usuario diseña la consulta y selecciona la opción "Ejecutar" para ver el resultado.	6. El sistema verifica que la consulta haya sido diseñada correctamente.
	7. El sistema muestra toda la información contenida en la base de datos relacionada con la consulta.
8. El usuario selecciona el fichero del cual desea ver la información que contiene. Ir al	

paso 9.	
Sección: "Consultar Vista Lógica de la Base de Datos"	
Acciones del Actor	**Respuesta del sistema**
5. El usuario selecciona la opción "Diseño de clases de la base de datos".	6. El sistema muestra la vista lógica de la base de datos.
Sección: "Consultar Fichero PDB"	
Acciones del Actor	**Respuesta del sistema**
1. El usuario indica el fichero PDB del cual desea consultar información en la opción "búsqueda rápida" la cual puede ser accedida desde cualquier página del sistema.	2. El sistema verifica que el identificador sea correcto.
	3. El sistema mostrará toda la información contenida en la base de datos relacionada con el fichero al que pertenece el identificador (Si el usuario inserta sólo una parte del identificador, el sistema mostrará la información contenida en la base de datos de todos los ficheros relacionados con la parte del identificador insertado).
4. El usuario selecciona el fichero del cual desea ver la información que contiene. Ir al paso 9.	
Sección: "Consultar Información de Fichero PDB"	
Acciones del Actor	**Respuesta del Sistema**
	9. El sistema muestra un listado de los ligandos y sitios de enlace del fichero PDB con la información contenida en la base de datos sobre los mismos.
Sección: "Consultar Información de Ligando"	
Acciones del Actor	**Respuesta del Sistema**
10. El usuario selecciona un ligando.	11. El sistema busca la información del ligando.

	12. El sistema muestra toda la información contenida en la base de datos relacionada con el ligando.
Sección: "Consultar Información de Sitio de Enlace"	
Acciones del Actor	**Respuesta del Sistema**
10. El usuario selecciona un sitio de enlace.	11. El sistema busca la información del sitio de enlace.
	12. El sistema muestra toda la información contenida en la base de datos relacionada con el sitio de enlace.
Flujos Alternos	
Sección: "Realizar Consulta Predeterminada a la Base de Datos"	
Acciones del Actor	**Respuesta del Sistema**
	5.1. Si la consulta no es correcta el sistema muestra el mensaje "Seleccione una opción válida". Ir al paso 2.
Sección: "Diseñar Nueva Consulta a la Base de Datos"	
Acciones del Actor	**Respuesta del Sistema**
	7.1. Si la consulta ya se encuentra insertada como predeterminada el sistema muestra el mensaje "La consulta se encuentra en el listado de consultas predeterminadas" y muestra el resultado.
	7.2. Si la consulta fue diseñada incorrectamente o no hay resultados el sistema muestra el mensaje "No se han obtenido resultados". Ir al paso 4.
	7.3. Si el usuario inserta los operadores "delete", "update", "create" o "select", el sistema muestra el mensaje "Los operadores delete, update, create y select no están permitidos". Ir al paso 4.
	7.4. Si el usuario hace una consulta sobre las tablas Cimuser o Comment el sistema muestra el mensaje "No se pueden efectuar consultas sobre esta tabla". Ir al paso 4.
Sección: "Consultar Fichero PDB"	

Acciones del Actor	Respuesta del Sistema
	3. Si el identificador es incorrecto el sistema muestra el mensaje " No ha habido ningún resultado para su búsqueda". El usuario puede volver al paso 1.
Sección: "Consultar Información de Fichero PDB"	
Acciones del Actor	**Respuesta del Sistema**
	9.1. Si el identificador es incorrecto el sistema muestra el mensaje "El identificador no existe".
Sección: "Consultar Información de Ligando"	
Acciones del Actor	**Respuesta del Sistema**
	12.1. Si el sistema no encuentra la información muestra el mensaje "El identificador no existe o no es un entero".
Sección: "Consultar Información de Sitio de Enlace"	
Acciones del Actor	**Respuesta del Sistema**
	12.1. Si el sistema no encuentra la información muestra el mensaje "El identificador no existe o no es un entero".
Prioridad:	Crítico

Descripción del Caso de Uso: Gestionar Consultas Predeterminadas.

Nombre del CUS	Gestionar Consultas Predeterminadas.
Actores	Administrador (Inicia el CU)
Propósito	Permitir insertar o eliminar consultas predeterminadas.
Resumen	El caso de uso se inicia cuando el administrador selecciona la opción "Insertar Consulta" o en la sección Administración, la opción "Listar consultas predeterminadas". El sistema muestra las interfaces correspondientes para dar cumplimiento al caso de uso. El administrador realiza las acciones necesarias. Finaliza el caso de uso.
Referencias	R4.2, R5.1, R5.2, R5.3.

Precondiciones	El administrador debe estar registrado en el sistema.
Poscondiciones	Quedan actualizadas las consultas predeterminadas.
Curso Normal de los Eventos	
Sección "Insertar Consulta Predeterminada"	
Acciones del Actor	**Respuesta del Sistema**
1. Después de haber diseñado una nueva consulta (Ir a Caso de Uso: Consultar Información de la Base de Datos, Sección: "Diseñar Nueva Consulta a la Base de Datos") el administrador selecciona la opción "Insertar Consulta".	2. El sistema muestra la interfaz que permite insertar un Título para la consulta.
3. El administrador inserta el título de la consulta.	4. El sistema verifica el título de la consulta.
	5. El sistema inserta la consulta como predeterminada y muestra la interfaz que contiene el listado de consultas predeterminadas actualizada con la nueva consulta insertada.
Sección "Listar Consultas Predeterminadas"	
Acciones del Actor	**Respuesta del Sistema**
1. El administrador selecciona la sección "Administración"	2. El sistema muestra la interfaz del área de administración con las posibles acciones a realizar por el administrador.
3. El administrador selecciona la opción "Listar consultas predeterminadas".	4. El sistema muestra el listado de consultas predeterminadas y brinda la posibilidad de eliminar consultas.
5. El administrador indica eliminar consulta.	6. El sistema verifica la consulta a eliminar.
	7. El sistema elimina la consulta predeterminada.
	8. El sistema muestra la lista de consultas predeterminadas actualizada.
"Flujos Alternos"	
Sección "Insertar Consulta Predeterminada"	

Acciones del Actor	Respuesta del Sistema
	5.1 Si el campo para insertar el titulo está vacío el sistema muestra un mensaje "Debe especificar un Título". Ir al paso 2.
	5.2 Si ocurre un error al insertar la consulta se muestra el mensaje "Se ha perdido la información".
	5.3 Si existe una consulta predeterminada con el mismo nombre el sistema muestra un mensaje "Una consulta con el mismo nombre ya ha sido insertada". Ir al paso 2.
	5.4 Si se transforma la url y la consulta no es válida se muestra el mensaje "Ha ocurrido un error en la consulta".
Sección "Listar Consultas Predeterminadas"	
Acciones del Actor	**Respuesta del Sistema**
	7.1. Si el sistema no encuentra la consulta a eliminar muestra el mensaje "El identificador no existe o no es un entero".
Prioridad	Crítico

Descripción del Caso de Uso: Exportar Resultado de Consulta.

Nombre del CUS	Exportar Resultado de Consulta (Caso de Uso Extendido).
Actores	Usuario (Inicia el CU)
Propósito	Exportar resultado de una consulta.
Resumen	El caso de uso se inicia cuando el usuario selecciona la opción "Exportar tar.gz", "Exportar ligand.pdb" o "Exportar contact.table". El sistema muestra una ventana para que el usuario seleccione la dirección donde desea exportar la consulta. El usuario selecciona la dirección, el sistema exporta la consulta. Finaliza el caso de uso.
Referencias	R 4.1, R 4.2, R4.4, R 4.5, R 6.1, R 6.2, R 6.3.
Precondiciones	El usuario debe estar registrado en el sistema.

Prioridad:	Crítico

Descripción del Caso de Uso: Actualizar la Base de Datos.

Nombre del CUS	Actualizar la Base de Datos.
Actores	Administrador (Inicia el CU).
Propósito	Actualizar la base de datos.
Resumen	El caso de uso se inicia cuando el administrador selecciona en la opción Procesar PDB una de la opciones “Almacenar un fichero PDB”, “Almacenar directorio de ficheros PDB” o “Almacenar ficheros PDB desde el servidor”. El administrador realiza las acciones necesarias. Finaliza el caso de uso.
Referencias	R 10.1, R 10.2, R 10.3.
Precondiciones	El administrador debe estar registrado en el sistema.
Poscondiciones	

Curso Normal de los Eventos	
Sección “General”	
1. El administrador selecciona la sección “Procesar PDB”.	2. El sistema muestra las opciones: “Almacenar un fichero PDB”, “Almacenar directorio de ficheros PDB” o “Almacenar ficheros PDB desde el servidor”.
Sección “Almacenar un fichero PDB”	
Acciones del Actor	**Respuesta del Sistema**
3. El administrador selecciona la opción “Almacenar un fichero PDB”.	4. El sistema comprueba el sistema operativo.
	5. El sistema muestra una ventana para que el administrador seleccione el fichero que desea almacenar.
6. El administrador selecciona la dirección del fichero a almacenar en la base de datos.	7. El sistema comprueba la dirección del fichero.
	8. El sistema envía a procesar el fichero.
	9. El sistema comprueba que los recursos para hacer

	el procesamiento hallan descargado correctamente en la PC cliente.
	10. El sistema realiza las reglas de extracción correspondientes.
	11. El sistema deposita el resultado de la extracción en un almacenamiento intermedio.
	12. El sistema obtiene el resultado de la extracción contenido en el almacenamiento intermedio.
	13. El sistema realiza las reglas de transformación.
	14. El sistema deposita el resultado de la transformación en un almacenamiento intermedio.
	15. El sistema obtiene el resultado de la transformación contenido en el almacenamiento intermedio.
	16. El sistema verifica la conexión con la base de datos.
	17. El sistema almacena el resultado en la base de datos.
	18. El sistema muestra el mensaje “El fichero fue almacenado en la base de datos correctamente”.
Sección “Almacenar directorio de ficheros PDB”	
Acciones del Actor	**Respuesta del Sistema**
3. El administrador selecciona la opción “Almacenar un directorio”.	4. El sistema comprueba el sistema operativo.
	5. El sistema muestra una ventana para que el administrador seleccione el directorio que desea almacenar.
6. El administrador selecciona el directorio.	7. El sistema comprueba la dirección del directorio.
	8. El sistema envía a procesar los ficheros contenidos en el directorio. Ir al paso 9.
Sección “Almacenar ficheros PDB desde el servidor FTP”	

Acciones del Actor	Respuesta del Sistema
3. El administrador selecciona la opción “Almacenar ficheros PDB desde el servidor FTP”.	4. El sistema comprueba el sistema operativo.
	5. El sistema envía a procesar los ficheros contenidos en el servidor FTP.
	6. El sistema comprueba que los recursos para procesar los ficheros se hallan descargado correctamente en la PC cliente.
	7. El sistema verifica la conexión con el servidor.
	8. El sistema se conecta con el servidor.
	9. El sistema comienza a descargar los ficheros. Ir al paso 10.
Flujos Alternos	
Sección “Almacenar un fichero PDB”	
Acciones del Actor	**Respuesta del Sistema**
	5.1. Si el sistema operativo no es Linux el sistema muestra el mensaje “El sistema operativo debe ser Linux”.
	8.1. Si la dirección del fichero es incorrecta el sistema muestra el mensaje “Debe seleccionar un fichero”.
	10.1. Si los recursos para hacer el procesamiento no se descargaron correctamente en la PC cliente el sistema muestra el mensaje “No se pudo copiar el directorio de trabajo correctamente”.
	13.1 Si el fichero no puede ser transformado el sistema mostrará un mensaje “El pdb (identificador del pdb) no es válido para aplicarle el algoritmo de trasformación”.
	17.1 Si falla la conexión con la base de datos el sistema muestra el mensaje “Se ha perdido la conexión con el servidor de base de datos”.

Sección "Almacenar directorio de ficheros PDB"	
Acciones del Actor	**Respuesta del Sistema**
	5.1. Si el sistema operativo no es Linux el sistema muestra el mensaje "El sistema operativo debe ser Linux".
	8.1 Si la dirección del directorio no es correcta el sistema muestra el mensaje "El directorio seleccionado no contiene ficheros pdb".
	13.1. Si existe algún problema cuando se están transformando los ficheros el sistema lo dejará registrado en un fichero de texto .log que se encontrará en la raíz del programa.
Sección "Almacenar ficheros PDB desde el servidor FTP"	
Acciones del Actor	**Respuesta del Sistema**
	5.1. Si el sistema operativo no es Linux el sistema muestra el mensaje "El sistema operativo debe ser Linux".
	Si los recursos para hacer el procesamiento no se descargaron correctamente en la PC cliente el sistema muestra el mensaje "No se pudo copiar el directorio de trabajo correctamente".
	8.1 Si la conexión es rechazada el sistema muestra el mensaje "Conexión rechazada revise el fichero srdpl_etl.xml".
	10.1 Si cuando se están descargando los ficheros del servidor falla la conexión el sistema muestra el mensaje "Se ha perdido la conexión con el banco pdb".
	13.1. Si existe algún problema cuando se están procesando los ficheros el sistema lo dejará registrado en un fichero de texto .log que se encontrará en la raíz

	del programa.
Prioridad:	Crítico

CONCLUSIONES

En este capítulo se definieron las características principales del sistema. Para dar cumplimiento a las funcionalidades y características del sistema se detectaron los requisitos funcionales y no funcionales; por lo que se determinó que para un mejor funcionamiento del sistema, quedaría compuesto por una aplicación escritorio y una aplicación Web. Se identificaron los actores del sistema así como los casos de uso del sistema con sus descripciones.

CAPÍTULO 3: DISEÑO DEL SISTEMA

INTRODUCCIÓN

En este capítulo se mostrarán los artefactos de diseño generados durante el flujo de trabajo de Análisis y Diseño para el sistema. Se describirá en términos del lenguaje de los programadores, cada una de las clases, con atributos y métodos, y la relación entre ellas. Esta descripción se encuentra detallada en las clases del diseño y los diagramas de secuencia generados para la aplicación de escritorio y la aplicación Web. Se realizará además el rediseño de la base de datos y se mostrará la vista lógica de la misma que cumple con los requerimientos actuales del CIM. Se presentará el modelo de despliegue para mostrar la distribución física del sistema.
La siguiente figura muestra cómo quedarán integradas ambas aplicaciones:

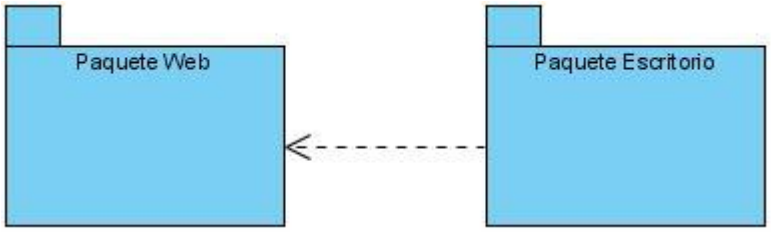


Fig. 4 Diagrama de Paquetes de la Solución

3.1 DISEÑO DE LA APLICACIÓN ESCRITORIO

La aplicación escritorio se definió con el objetivo de realizar el proceso de recopilación de datos provenientes de la fuente Protein Data Bank para almacenarlos en una base de datos. Los artefactos generados incorporan el uso de los procesos ETL para representar el correcto manejo del gran volumen de datos a extraer, transformar y cargar en la base de datos. En la implementación del módulo escritorio se introduce la arquitectura en tres capas, por lo que en los diagramas de las clases del diseño se refleja este patrón arquitectónico (Ver fig. 5). Los diagramas de interacción y los diagramas de las clases del diseño muestran además cómo será utilizado el framework spring en la fase de implementación.

Fig. 5 Patrón Arquitectónico Tres Capas

3.1.1 Aplicación de los Patrones de Diseño

- ✓ Patrón Alta Cohesión

Se utiliza por ejemplo en la creación de la interfaz ITransform y su implementación PdbTransformDataImpl que son las que tienen la responsabilidad de manejar toda la información referente al proceso de transformación de los archivos PDB, de esta forma se logra abstraer el servicio TransformService de la forma en que se hace dicho proceso, así mismo ocurre para los servicios de extracción y carga. La siguiente figura evidencia el uso de este patrón.

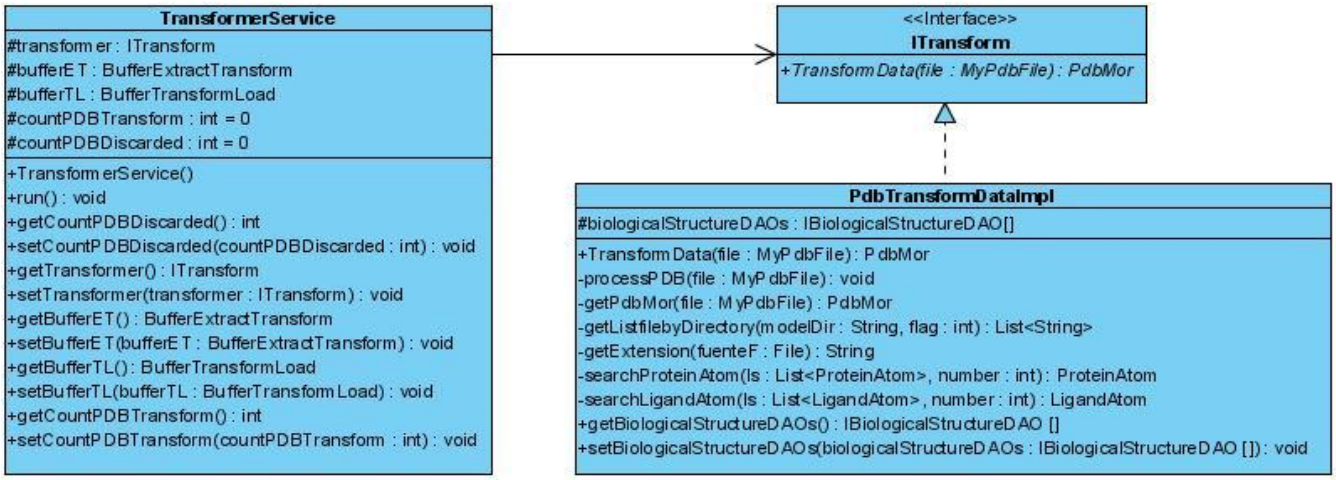


Fig. 6 Patrón Alta Cohesión

- ✓ Patrón Bajo Acoplamiento

Se logró un bajo acoplamiento de clases al utilizar el patrón inyección de dependencias. Un conjunto de clases importan determinadas interfaces y no crean directamente instancias de las implementaciones de dichas interfaces. Las instancias de dichas interfaces se configuran en un fichero

XML logrando un acoplamiento o desacoplamiento sin necesidad de modificar el código fuente. Por ejemplo las interfaces IExtract, ITransform e ILoad y sus respectivas implementaciones

Fig. 7 Patrón Bajo Acoplamiento

- ✓ Patrón Puente

Como se muestra en la siguiente figura en la aplicación escritorio se pone en práctica este patrón para separar la interfaz IBiologicalStructureDAO de su implementación, en este caso es implementada por las clases LigandsCIMDAOImpl, TableCIMDAOImpl y MorCIMDAOImpl.

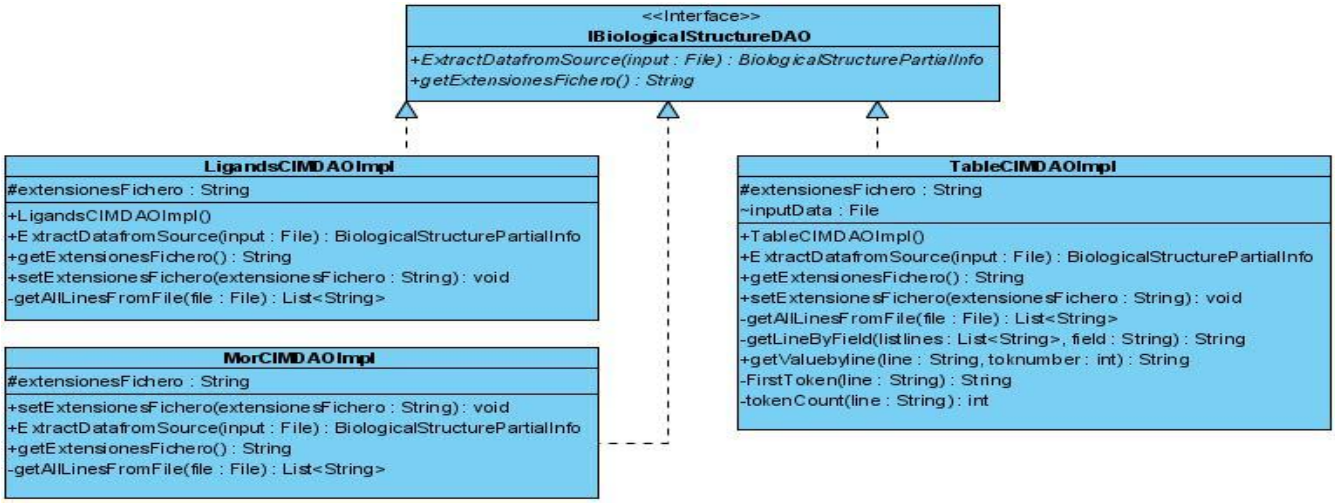


Fig. 8 Patrón Puente

- ✓ Patrón Factory

En el diseño de la capa de datos se ha utilizado el patrón Factory para la creación de las clases que contienen toda la información para la conexión a la base de datos a través de los parámetros obtenidos del fichero de configuración srdpl-daoContext.xml utilizando inyección de dependencias internamente al invocar al fichero de configuración hibernate.properties donde se encuentran los atributos para la conexión. De esta manera se obtiene un código fuente mucho más legible y organizado debido a que toda la configuración de la capa de datos se encuentra centralizada. Este ejemplo se aplica al diseño de ambas aplicaciones. Ver fig. 10

- ✓ Patrón Instancia Única

En la aplicación escritorio se garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. Todas estas ventajas son introducidas por el framework spring, ejemplo de ello son los archivos de configuración srdpl-daoContext.xml donde se encuentran los beans de los objetos de acceso a datos, los cuales siempre estarán disponibles a través de la clase MyBeanFactory. Este ejemplo se aplica al diseño de ambas aplicaciones. Ver fig. 10

- ✓ Patrón DAO

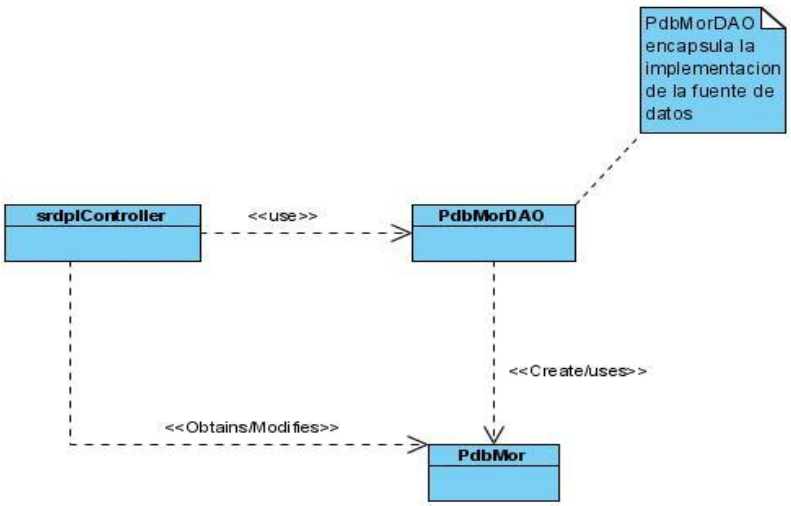


Fig. 9 Patrón DAO Escritorio

En la aplicación escritorio la clase PdbMorDAO es la encargada de utilizar los objetos de la clase entidad PdbMor que son llamados desde el controlador srdplControler. El controlador srdplController

es la clase que quiere acceder a la fuente de datos para poder almacenar o consultar datos. PdbMorDAO abstrae al srdplControler de los detalles del acceso a la fuente de datos. PdbMor es la clase intermedia entre el srdplControler y la clase de acceso a datos PdbMorDAO.

- ✓ Patrón Inyección de dependencias.

El patrón inyección de dependencias se utiliza para la configuración y utilización de los objetos de acceso a datos removiendo de esta manera del código los elementos de configuración como: la dirección del servidor de base de datos, el puerto por el que se establecerá la conexión, la cantidad total de conexiones, el usuario y se realiza de manera transparente, pues se ejecuta enteramente a través de Spring (MyBeanFactory es la clase que se encarga de construir los beans del contexto definido). Este ejemplo se aplica al diseño de ambas aplicaciones.

Fig. 10 Patrón Inyección de Dependencias

- ✓ Patrón Facade

Como se muestra en la siguiente figura se diseñaron los servicios ExtractorService, TransformerService y LoaderService para la implementación de las reglas de negocio y para ocultar la complejidad de las clases inferiores a la clase controladora srdplController cuando los solicita.

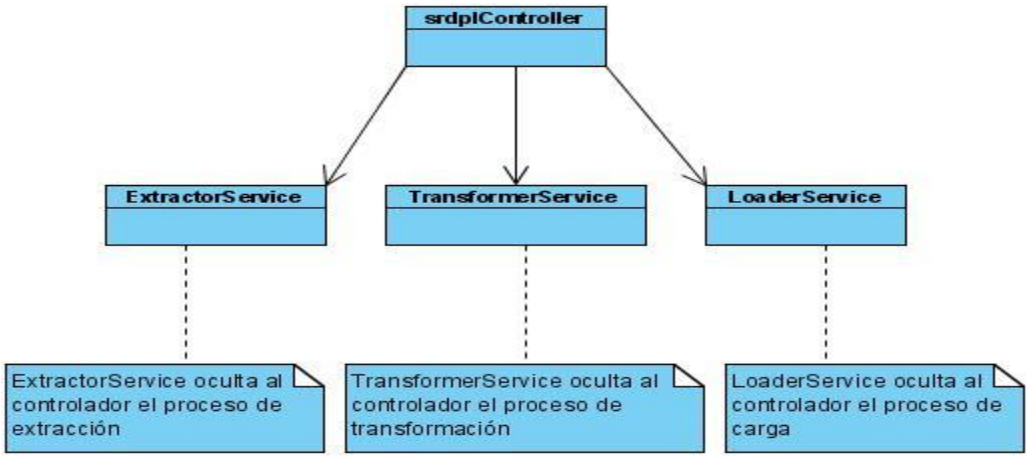


Fig. 11 Patrón Facade

3.1.2 Diagramas de Clases del Diseño

La siguiente figura muestra el diseño de los paquetes principales para la aplicación escritorio cumpliendo con el patrón arquitectónico definido. El administrador realiza las peticiones a la clase principal contenidas en el paquete visual, las cuales son atendidas por la clase controladora del paquete controller, esta clase es la encargada de ejecutar los procesos de extracción, transformación y carga para los archivos PDB a almacenar en la base de datos. En el paquete extract se encuentran las clases encargadas de obtener los ficheros y aplicarles las reglas definidas para el proceso de extracción. El resultado de la extracción es depositado en un buffer intermedio que se encuentra en el paquete útil el cual es el punto de partida del proceso de transformación. En el paquete transform se encuentran las clases encargadas de aplicar las reglas definidas para el proceso de transformación y el resultado es depositado en otro buffer intermedio. Finalmente en el paquete load se encuentran las clases encargadas de obtener los datos depurados que se encuentran en el buffer intermedio y cargarlos a la base de datos. La capa de acceso a datos, que se encuentra representada por el paquete dao, encapsula el proceso de carga de los datos en la base de datos.

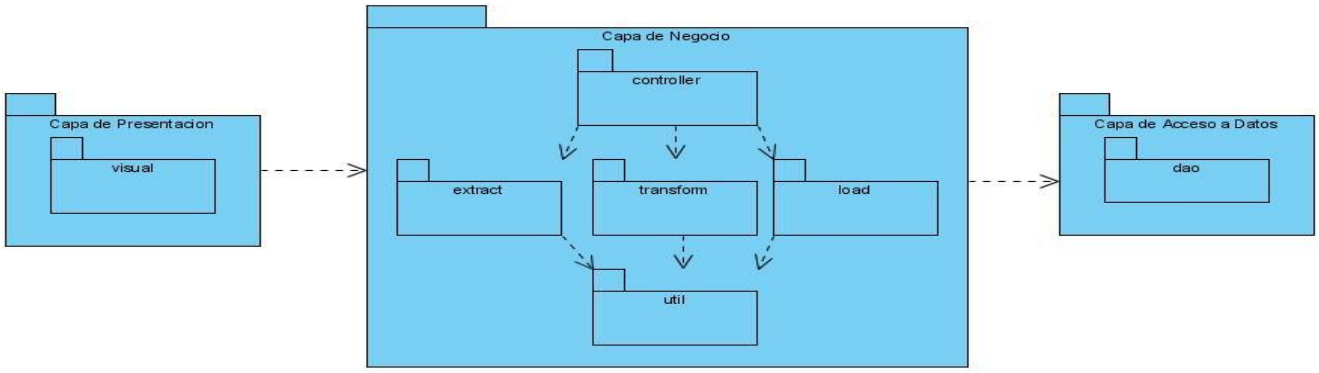


Fig. 12 Diagrama General Aplicación Escritorio

Cada proceso de extracción, transformación y carga de datos será atendido por una clase de servicio ExtractorService, TransformerServie y LoaderService que se encuentran en los paquetes extract, transform y load respectivamente. Estas clases heredan de la clase Thread, por lo que cada una se hallará verificando constantemente si en los almacenamientos intermedios se encuentran datos para procesar, de ser así el servicio que encuentre información comenzará a procesarla independientemente de las acciones de otro servicio.

Los siguientes diagramas muestran con un nivel de profundización mayor los paquetes y clases que contienen los paquetes Extract, Transform y Load.

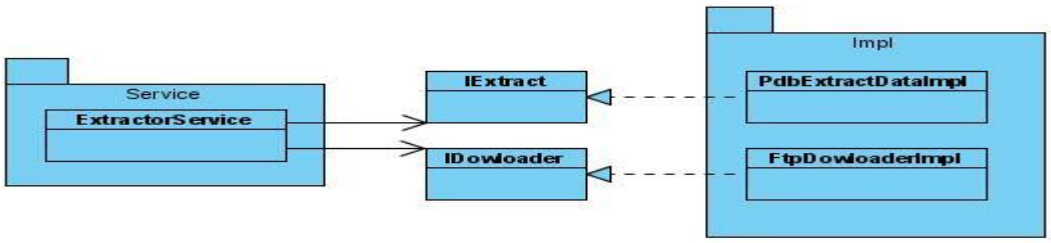


Fig. 13 Paquete extract

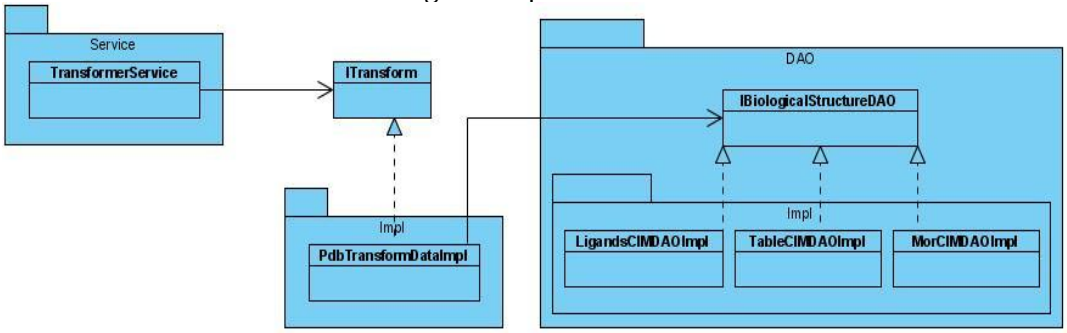


Fig. 14 Paquete transform

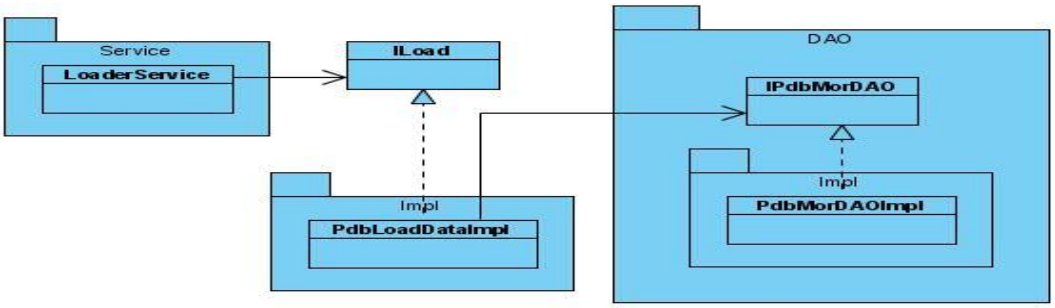


Fig. 15 Paquete load

A continuación se explica en detalle cómo procede cada clase definida en el flujo de trabajo de Análisis y Diseño, en el proceso de almacenamiento de los archivos provenientes de la fuente Protein Data Bank. La figura muestra el diseño de las clases que representan el paquete Visual donde el administrador, después de autenticarse en la ventana VLoggin, realiza las distintas peticiones de almacenamiento de los ficheros. Las peticiones del administrador son realizadas a la clase VPrincipal y la clase VProgress muestra en todo momento el avance en el proceso de almacenamiento de los archivos PDB, mientras que la clase updateProgressBar, como su nombre lo indica, mantiene la clase VProgress actualizada.

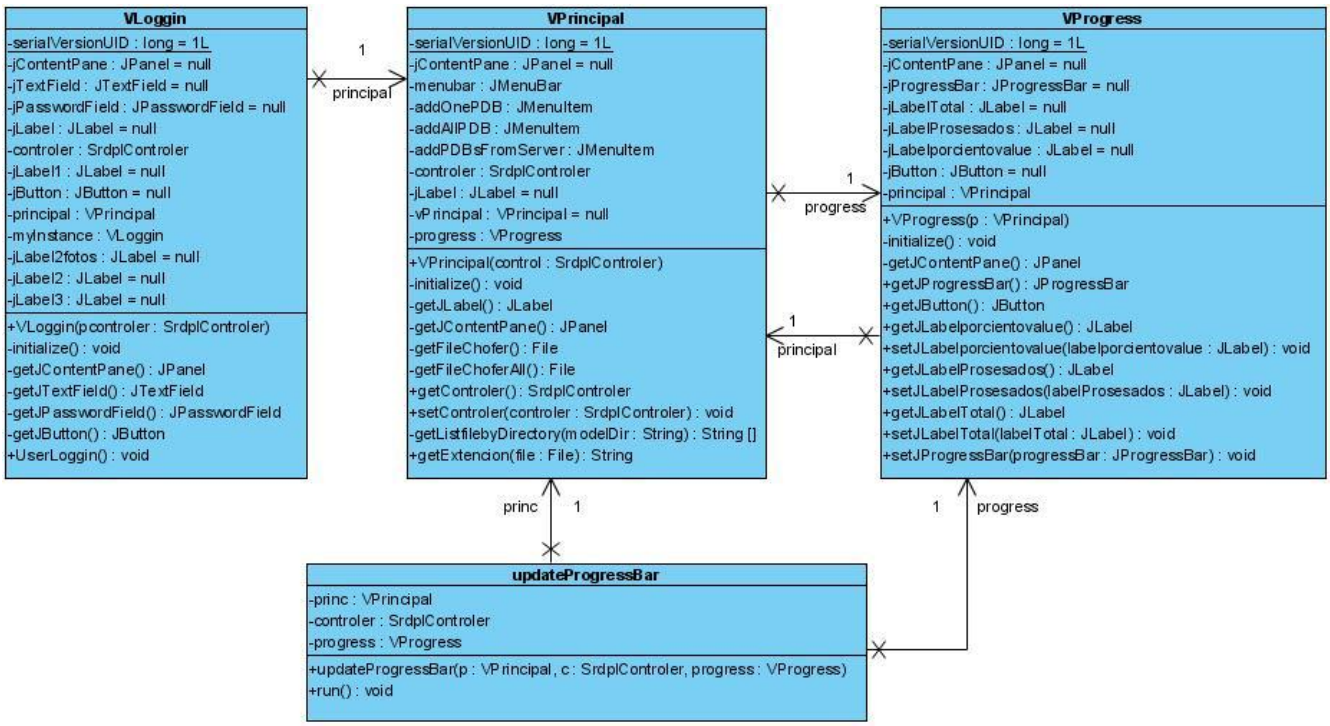


Fig. 16 Paquete visual

La siguiente figura muestra la clase controladora que atiende las peticiones de la clase VPrincipal y ejecuta los procesos de extracción, transformación y carga de los archivos PDB a la base de datos.

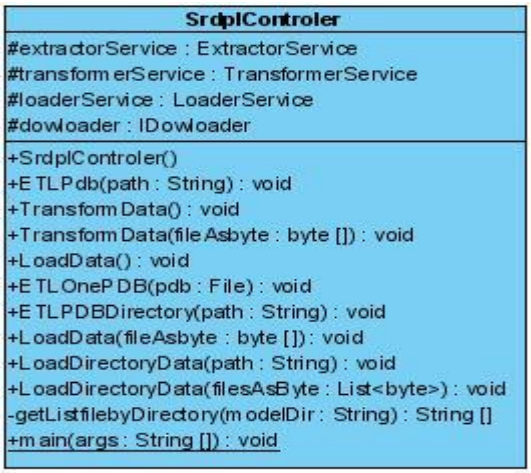


Fig. 17 Paquete controller

3.1.2.1 PROCESO DE EXTRACCIÓN DE DATOS

A continuación se explica las interacciones de las clases que intervienen en el proceso de extracción de los archivos PDB. La clase ExtractorService comprueba si la petición del administrador es almacenar los ficheros PDB desde el servidor, de ser así la interfaz IDownloader y su implementación son las encargadas de crear la conexión al servidor y comenzar a descargar los ficheros en cantidades de 50. Las reglas de extracción se realizan en la clase IExtract y su implementación PdbExtractDataImpl. Los ficheros obtenidos de la extracción son almenados en el BufferExtractTransform.

En caso de que sea un directorio o un fichero lo que se desee almacenar se comprueba que sea un directorio o un fichero respectivamente y se hace el mismo proceso obviando el uso de la interfaz IDownloader y su implementación.

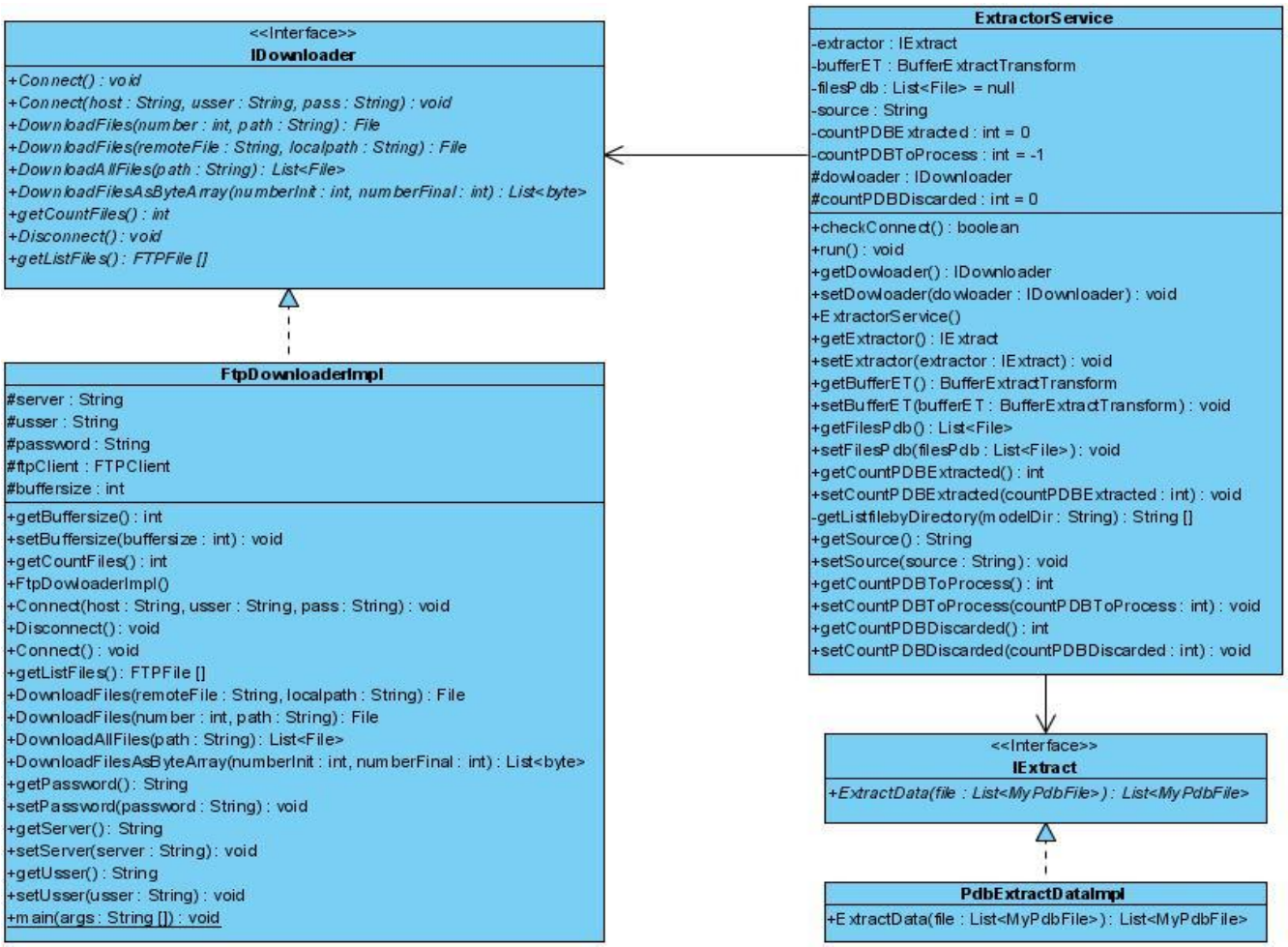


Fig. 18 Relaciones entre clases del paquete extract

3.1.2.2 PROCCESO DE TRANSFORMACIÓN DE DATOS

Este proceso es el que lleva a cabo la transformación de los ficheros por medio del algoritmo diseñado por los investigadores del CIM. A continuación se explicará cómo se hace dicho proceso.

En el proceso de transformación la clase de servicio TransformerServie obtiene los datos almacenados en el BufferExtractTransform y delega a la interfaz ITransform y su implementación PdbTransformDataImpl el procesamiento de los mismos. Primeramente el o los ficheros PDB son transformados por el algoritmo diseñado por los investigadores del CIM, el resultado será una copia en disco duro de ficheros ligand.pdb, contact.table y .mor por cada ligando encontrado, los que serán leídos a través de las clases LigandsCIMDAOImpl, TableCIMDAOImpl y MorCIMDAOImpl en dependencia del formato de los mismos. Con la información obtenida se creará una lista de objetos de tipo BiologicalStructurePartialInfo que pueden ser TableCimStructure, LigandCIMStructure y MorCIMStructure (representan las principales clases del modelo del módulo escritorio). Por cada trío

de objetos TableCimStructure, LigandCIMStructure y MorCIMStructure se creará uno de tipo PdbMor que tendrá la información necesaria para guardar una tupla en la base de datos. El resultado del proceso será almacenado en el BufferTransformLoad.

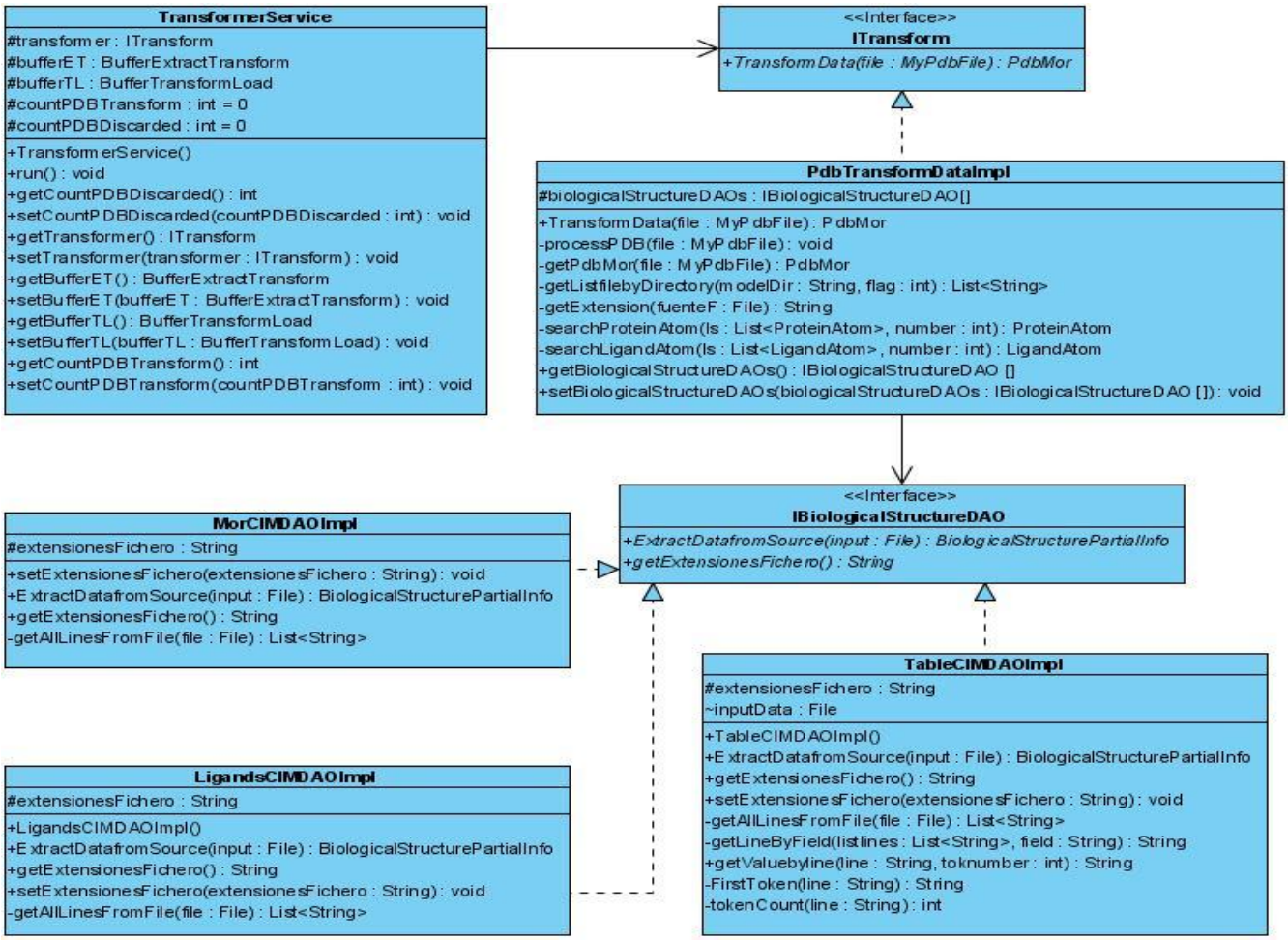


Fig. 19 Relaciones entre clases del paquete transform

3.1.2.3 PROCESO DE CARGA DE DATOS

En el proceso de carga de datos la clase de servicio LoaderService obtiene los datos que se encuentran almacenados en el BufferTransformLoad y delega a la interfaz ILoad y su implementación PDBLoadDataImpl el proceso de carga de los mismos a la base de datos. Lo que se almacena en la base de datos es un objeto PdbMor mediante un objeto PdbMorDAO de acceso a datos.

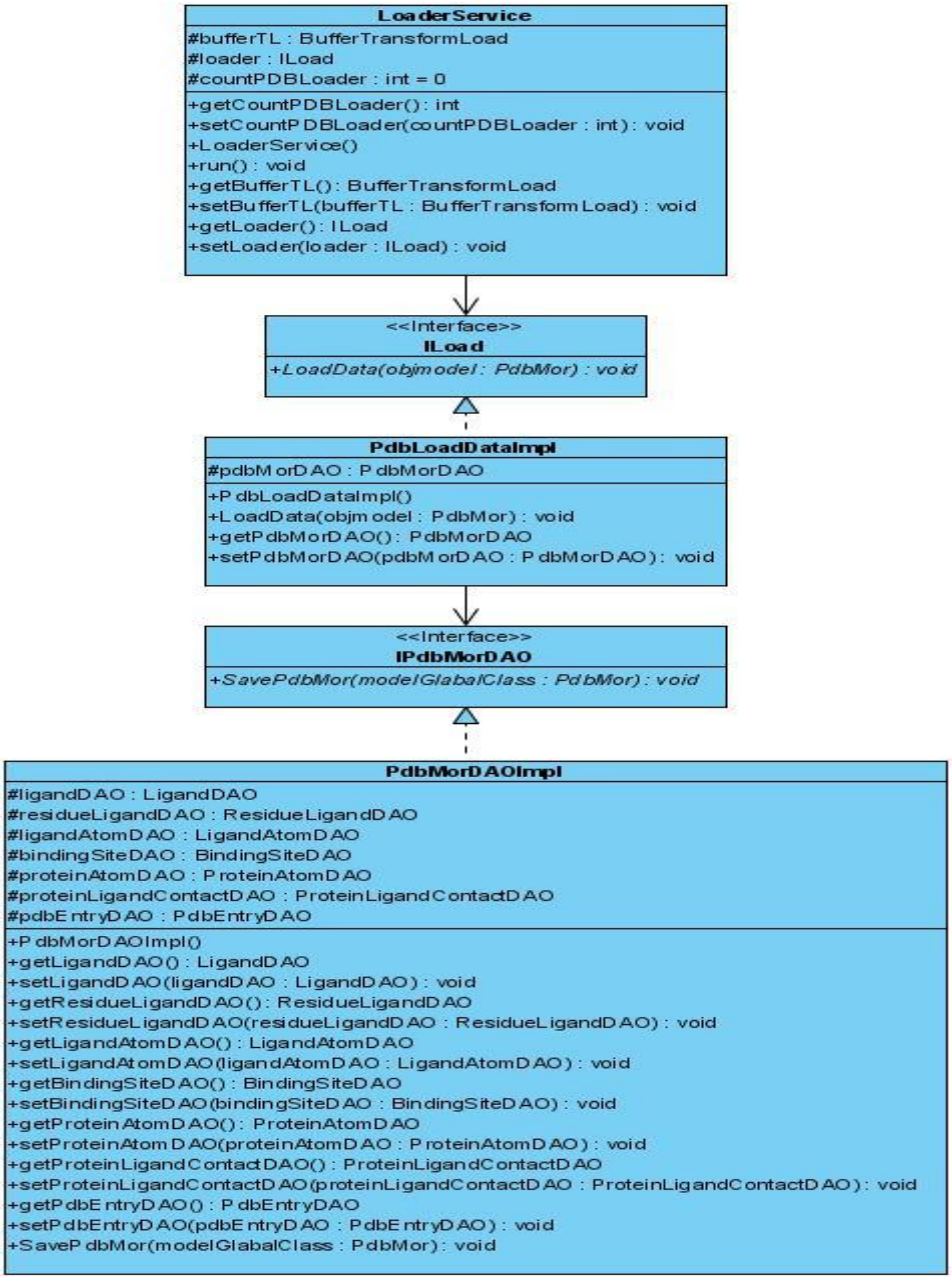


Fig. 20 Relaciones entre clases del paquete load

La siguiente figura muestra las clases contenidas en el paquete útil. Las clases BufferExtractTransform y BufferTransformLoad son los almacenamientos intermedios entre el proceso de extracción y transformación y el proceso de transformación y carga respectivamente. La clase PdbMor es una clase entidad que se crea en memoria y se utiliza en el proceso de carga de los datos a la base de datos, esta clase tendrá la información necesaria para guardar una tupla en la base de datos.

Fig. 21 Paquete útil

3.1.3 Diagramas de Secuencia

Los diagramas de secuencia muestran en detalle cómo interactúan las clases y guían al programador durante la fase de implementación.

La figura 22 expone la interacción entre las clases que componen la aplicación escritorio cuando el administrador indica almacenar un fichero PDB. La figura 23 muestra la interacción entre las clases que componen el módulo escritorio cuando el administrador indica almacenar un directorio de ficheros PDB y los mismo ocurre en la figura 24 cuando el administrador selecciona almacenar los ficheros PDB de el servidor FTP.

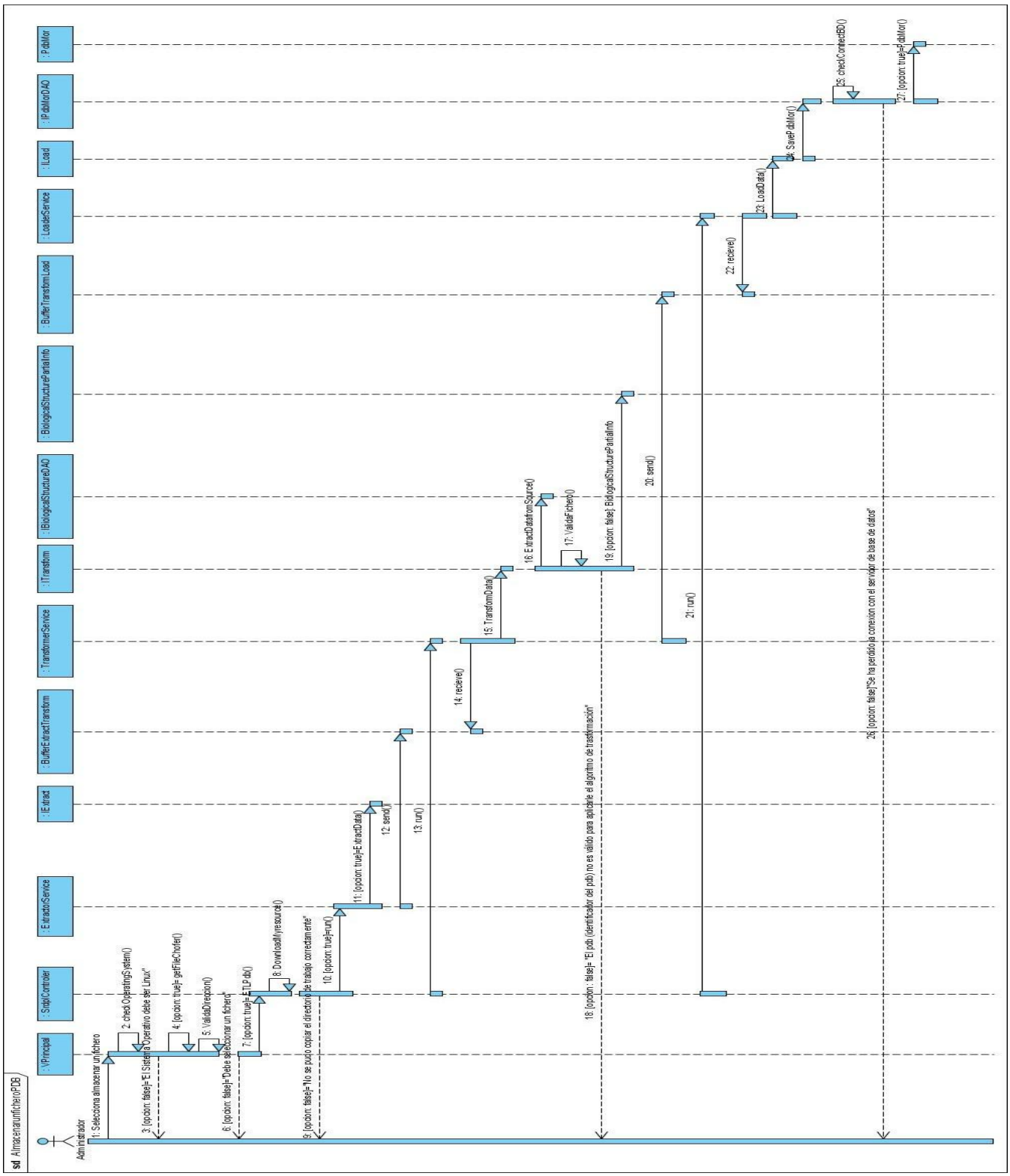


Fig. 22 DSCU Actualizar la Base de Datos (Escenario Almacenar un fichero PDB)

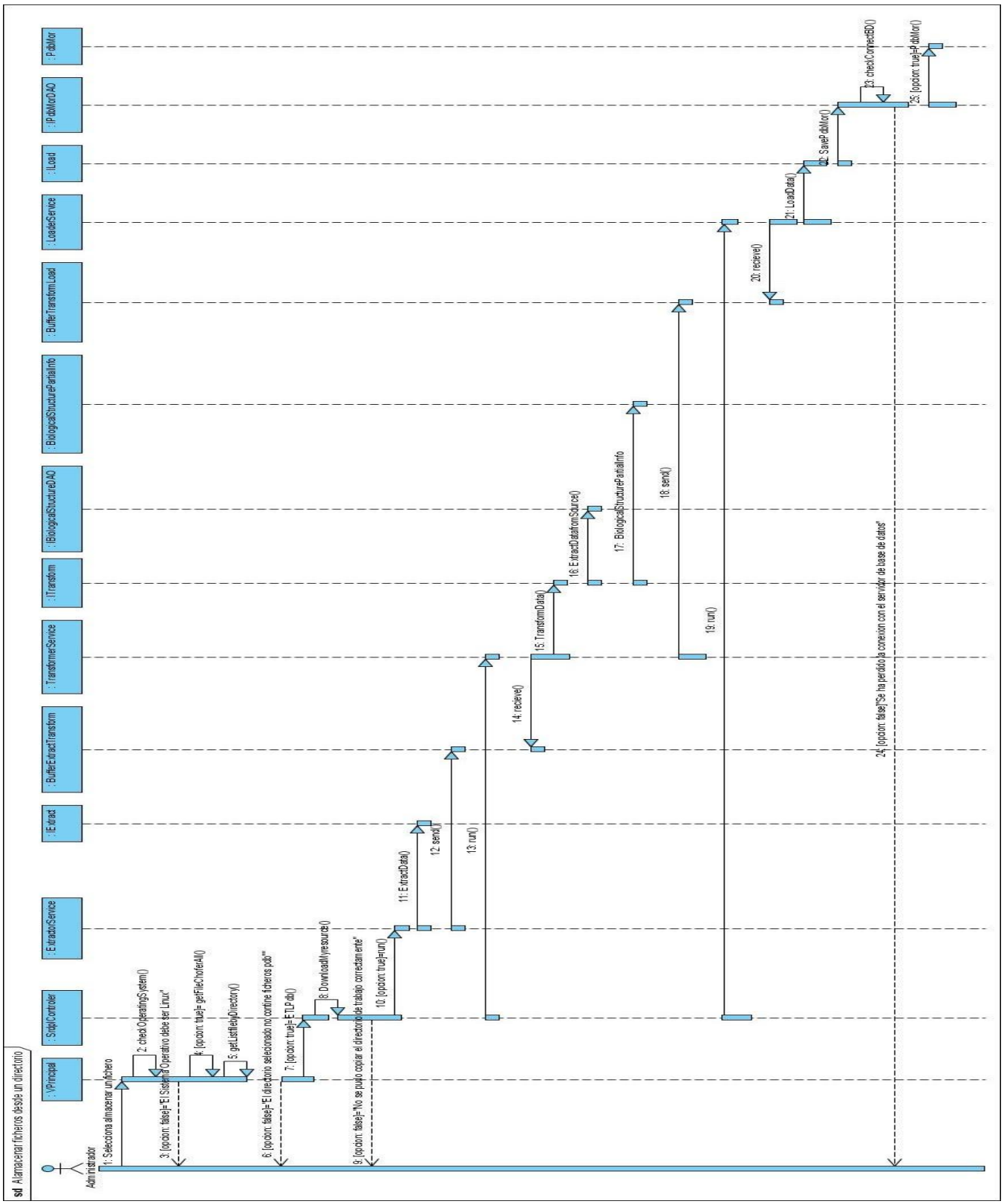


Fig. 23 DSCU Actualizar la Base de Datos (Escenario Almacenar directorio de ficheros PDB)

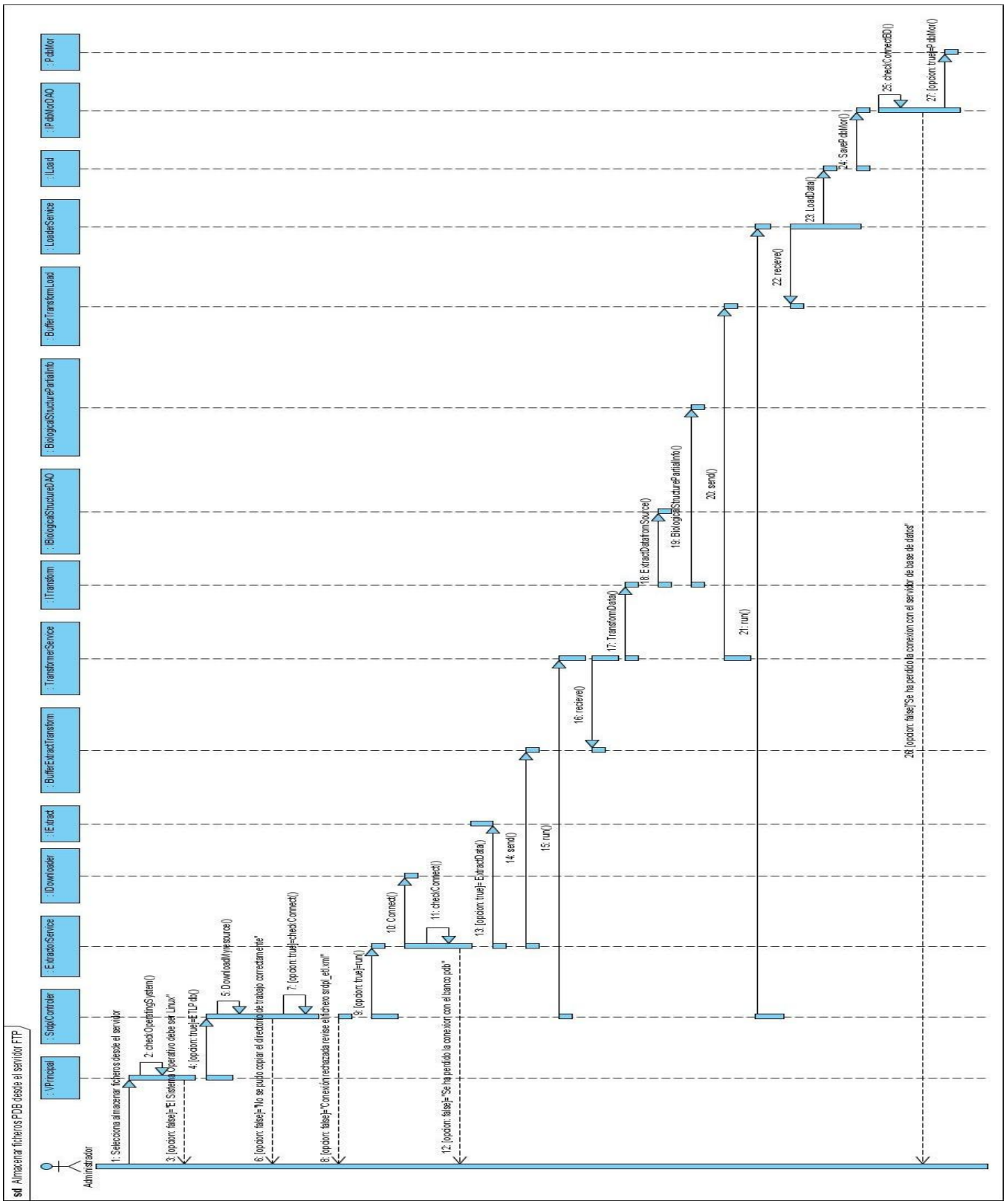


Fig. 24 Fig. 21 DSCU Actualizar la Base de Datos (Escenario Almacenar ficheros PDB desde el servidor FTP)

3.2 DISEÑO DE LA APLICACIÓN WEB

El diseño de la aplicación Web se definió para que los usuarios pudieran consultar la base de datos. El uso del framework spring en la implementación del sistema introduce el patrón arquitectónico Modelo Vista Controlador (MVC), por lo que en los diagramas de clases del diseño de la aplicación Web se reflejan las interacciones entre los principales elementos cumpliendo con el patrón arquitectónico definido.

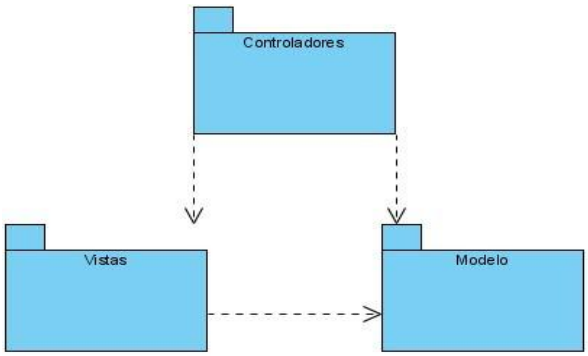


Fig. 25 Patrón Arquitectónico Modelo Vista Controlador

3.2.1 Aplicación de los Patrones de Diseño

- ✓ Patrón Alta Cohesión

Se utiliza en la creación de las clases controladoras, estas son las encargadas de controlar el envío hacia las vistas de la información que los servicios le brindan, y los servicios obtienen la información a través de las clases de acceso a datos.

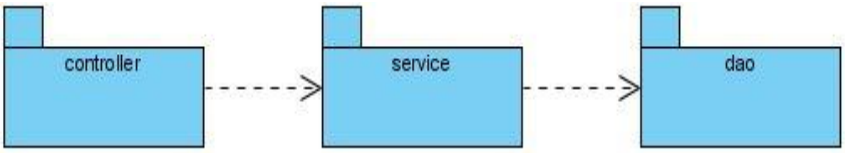


Fig. 26 Patrón Alta Cohesión

- ✓ Patrón Bajo Acoplamiento

Se logró bajo acoplamiento en las clases al utilizar el patrón inyección de dependencias. Un conjunto de clases como los servicios y las clases de acceso a datos importan determinadas interfaces y no crean directamente instancias de las implementaciones de dichas interfaces. Las instancias de dichas interfaces se configuran en un fichero XML logrando un acoplamiento o desacoplamiento sin necesidad de modificar el código fuente. Ver fig. 27

- ✓ Patrón Puente

En la implementación del módulo web se pone en práctica para separar las interfaces de los servicios de su implementación así como las interfaces de las clases de acceso a datos de su implementación.

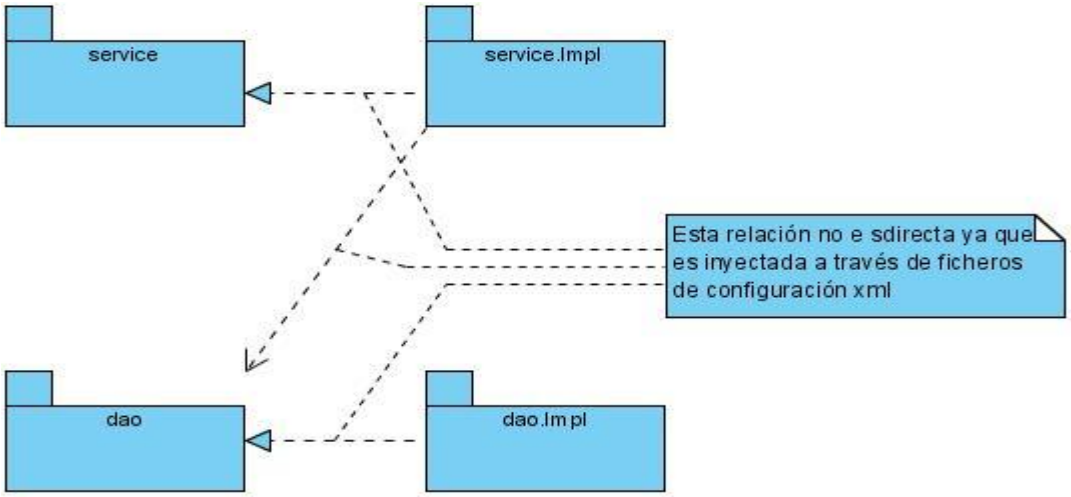


Fig. 27 Patrón Puente

- ✓ Patrón DAO

En la aplicación web las interfaces del paquete dao y sus respectivas implementaciones son las encargadas de crear y utilizar los objetos de las clases del paquete model que son llamadas desde los controladores. Los controladores son los que quieren acceder a la fuente de datos para poder almacenar o consultar datos. Las clases de acceso a datos abstraen los controladores de los detalles del acceso a la fuente de datos. Las clases del paquete model son las intermediarias entre los controladores y las clases de acceso a datos.

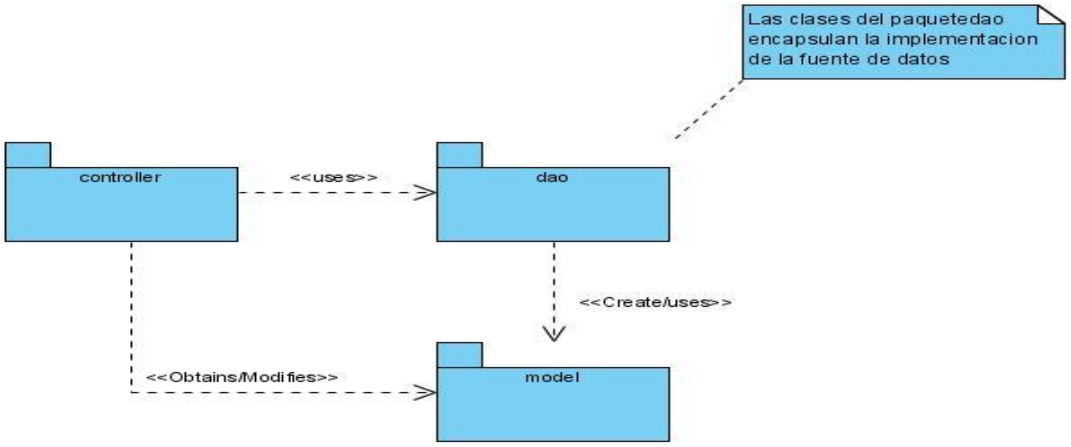


Fig. 28 Patrón DAO

- ✓ Patrón Facade

Se diseñaron varios servicios para la implementación de las reglas de negocio y para ocultar la complejidad de las clases inferiores a aquellas que solicitan los servicios. Ver fig. 27

3.2.2 Diagramas de Clases del Diseño

En las clases del diseño las peticiones realizadas por los usuarios en las páginas clientes serán atendidas por el controlador frontal srdpl_servlet quien delegará a un componente de Spring el procesamiento de la solicitud para determinar el controlador especializado en atender la misma. Este controlador será el encargado de realizar las peticiones necesarias a las clases que prestan servicios, a las clases de acceso a datos y a las clases entidades del modelo. Luego notificará al controlador frontal quien delegará a otro componente de Spring la construcción de las páginas clientes con los datos requeridos por el usuario.

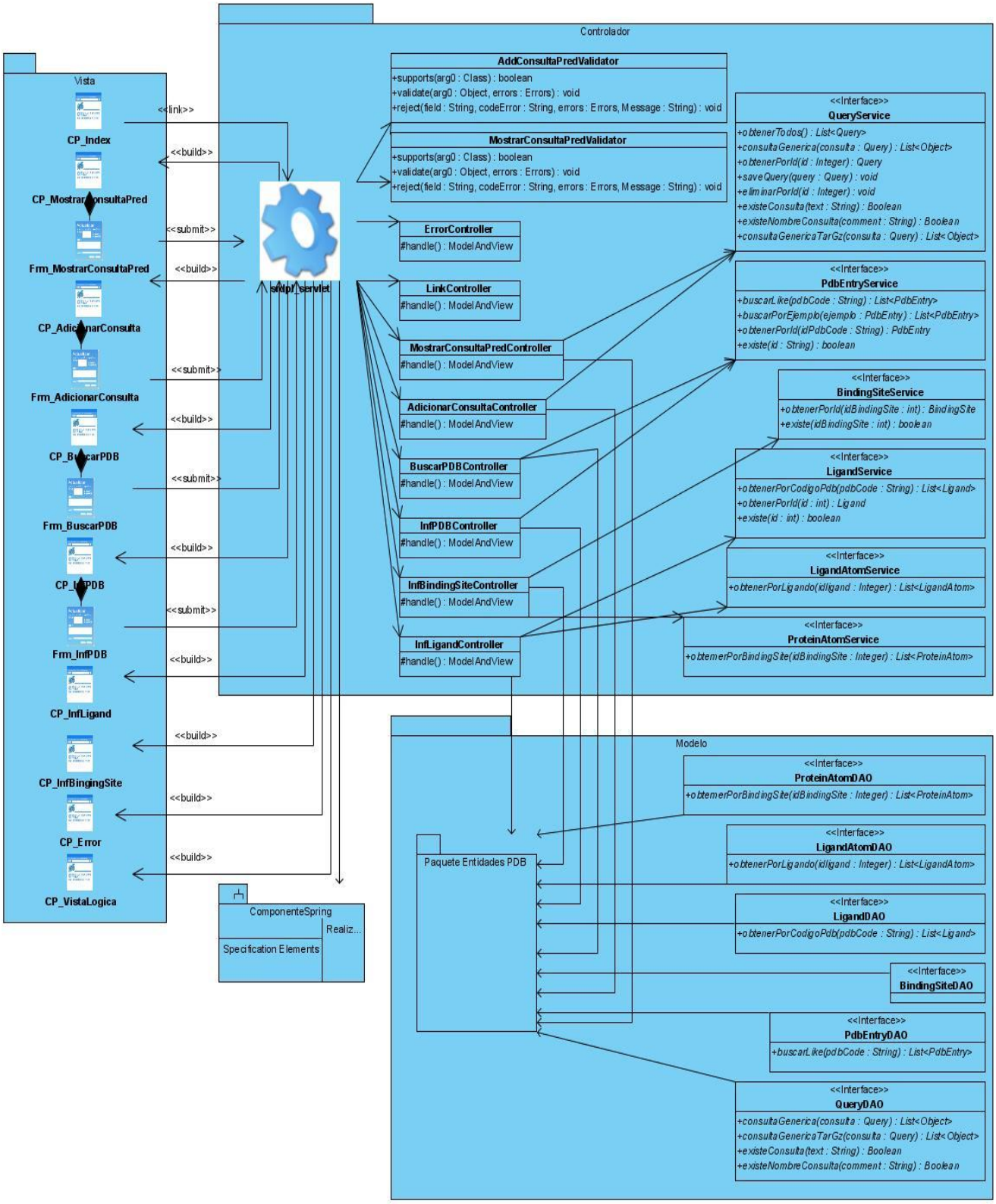


Fig. 29 DCDCU Consultar Información de la Base de Datos

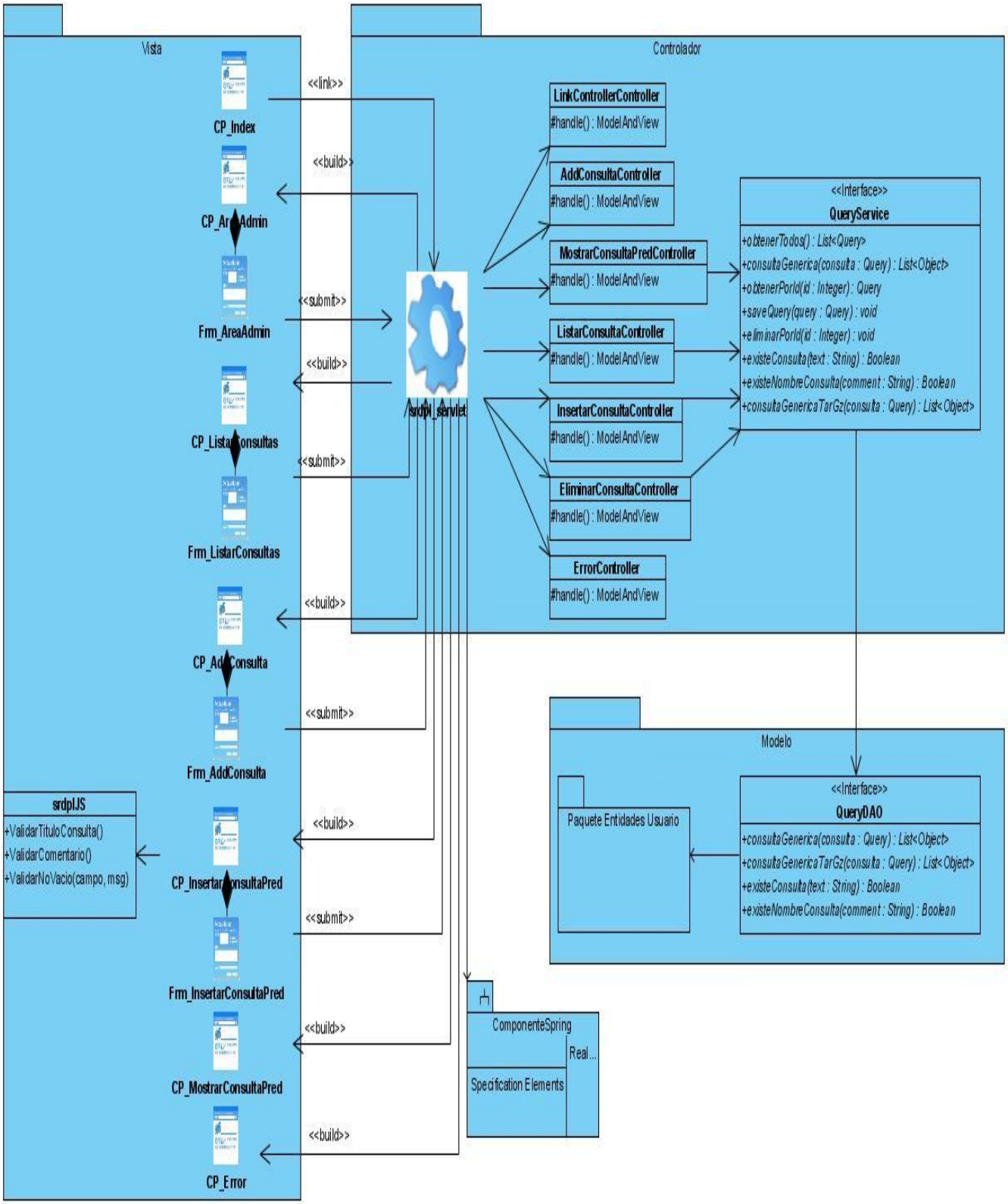


Fig. 30 DCDCU Gestionar Consultas Predeterminadas

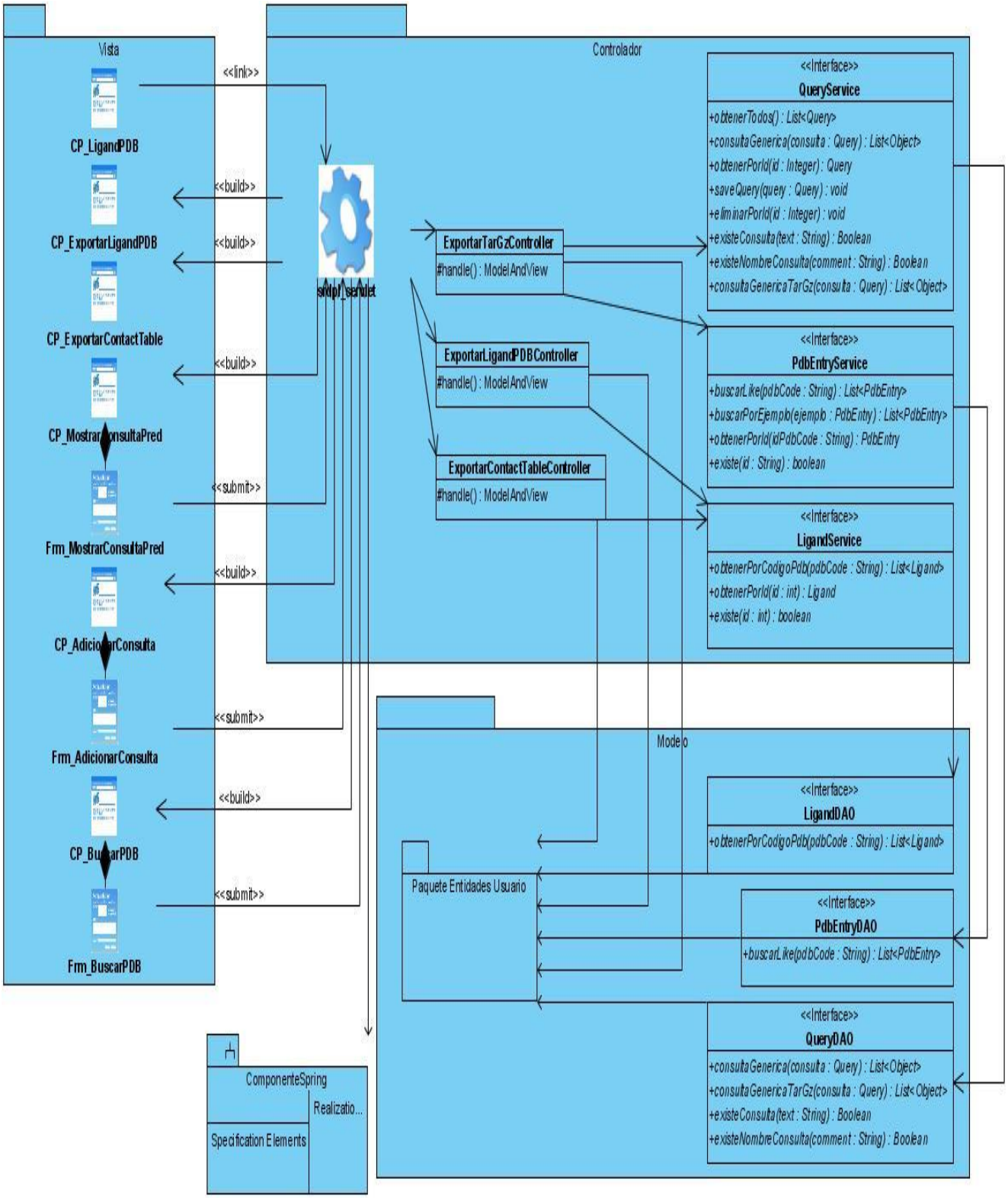


Fig. 31 DCDCU Exportar Resultado de Consulta

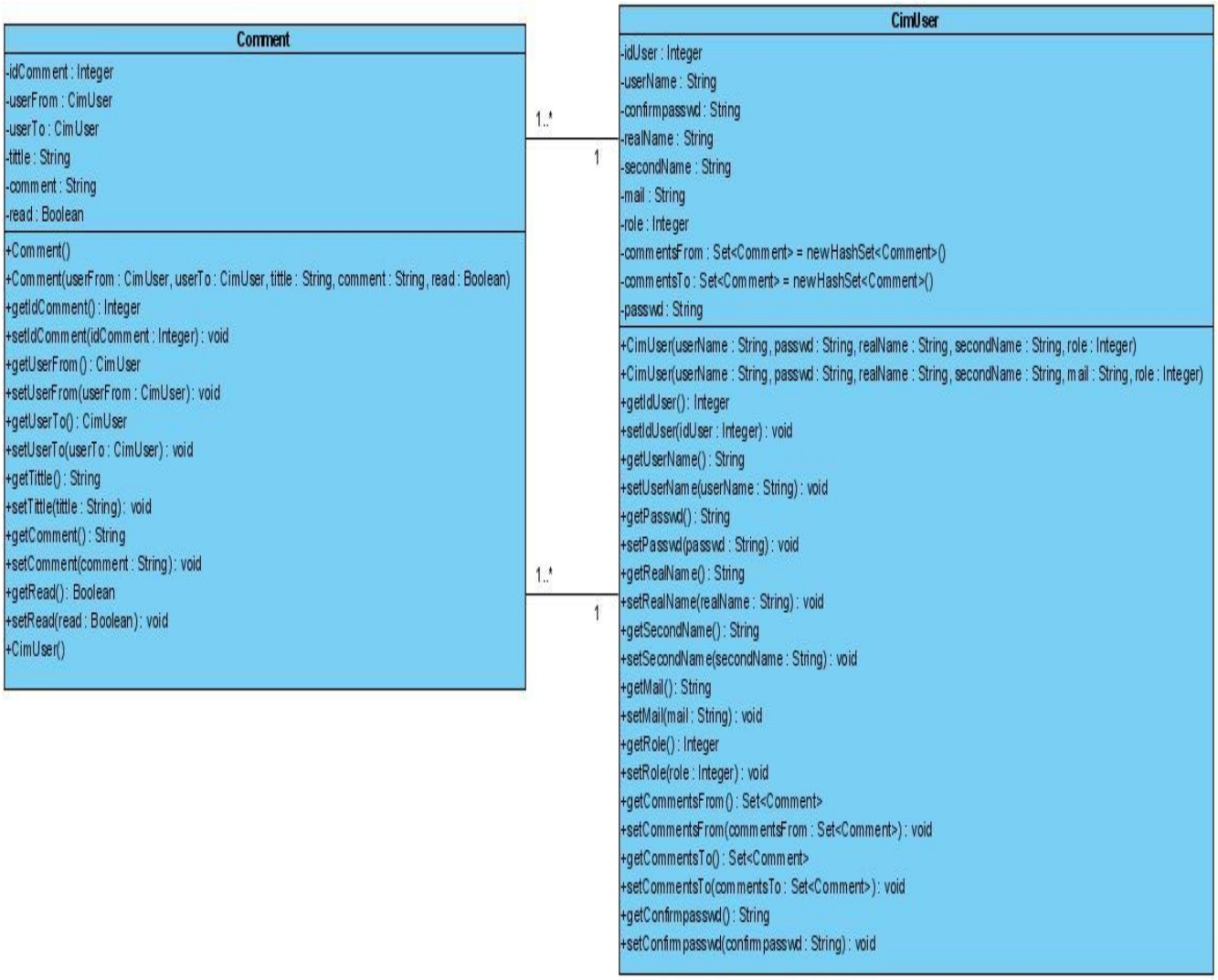


Fig. 32 Paquete Entidades Usuario

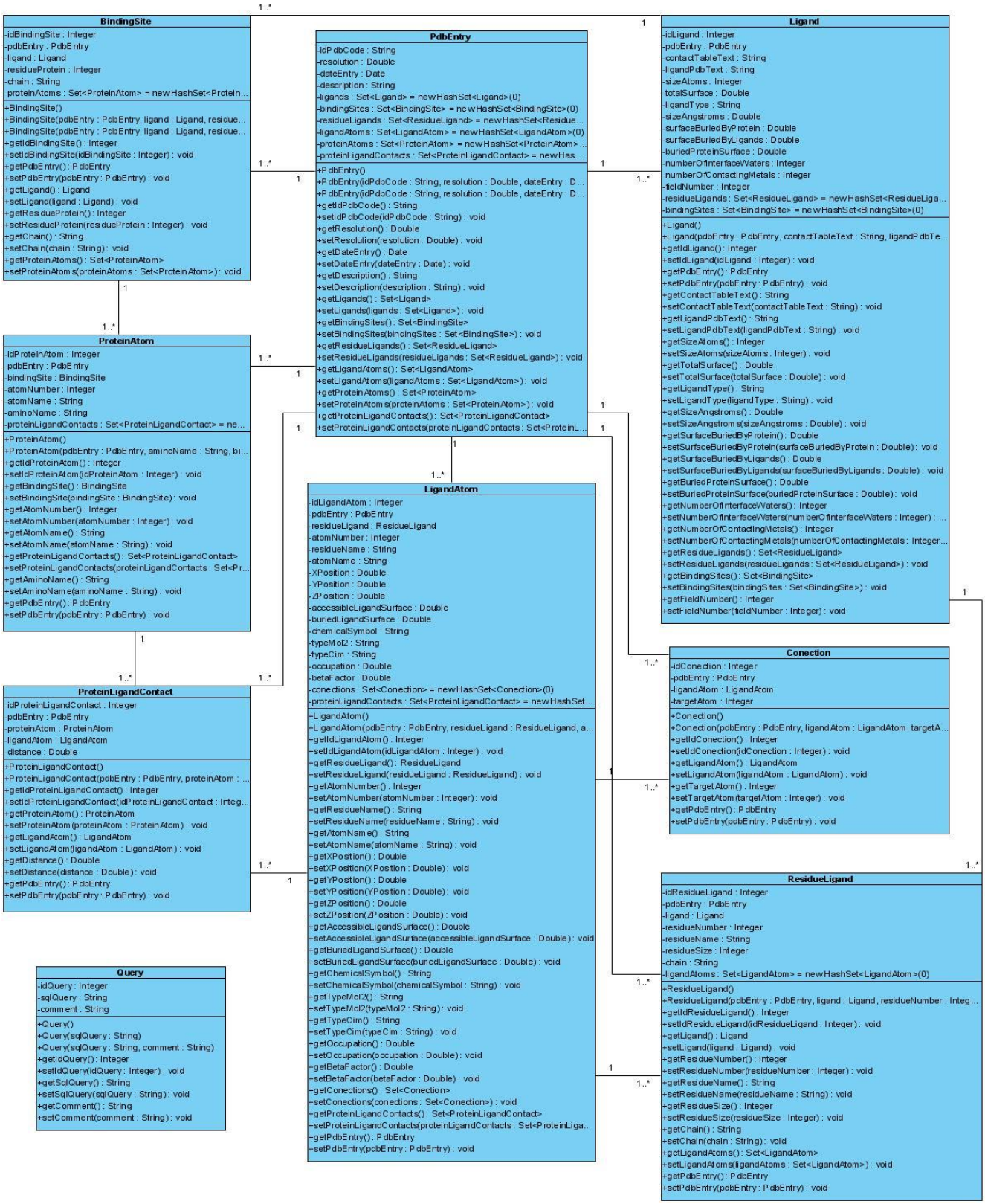


Fig. 33 Paquete Entidades PDB

3.2.3 Diagramas de Secuencia

Los diagramas de secuencia muestran en detalle las interacciones entre las clases y son la guía de los programadores en la fase de implementación.

Caso de Uso Consultar Información de la Base de Datos.

- ✓ Escenario Realizar Consulta Predeterminada a la Base de Datos.

La Fig. 34 muestra las interacciones que se producen entre las clases que son necesarias para que el sistema muestre el resultado de una consulta predeterminada al usuario que realiza la petición.

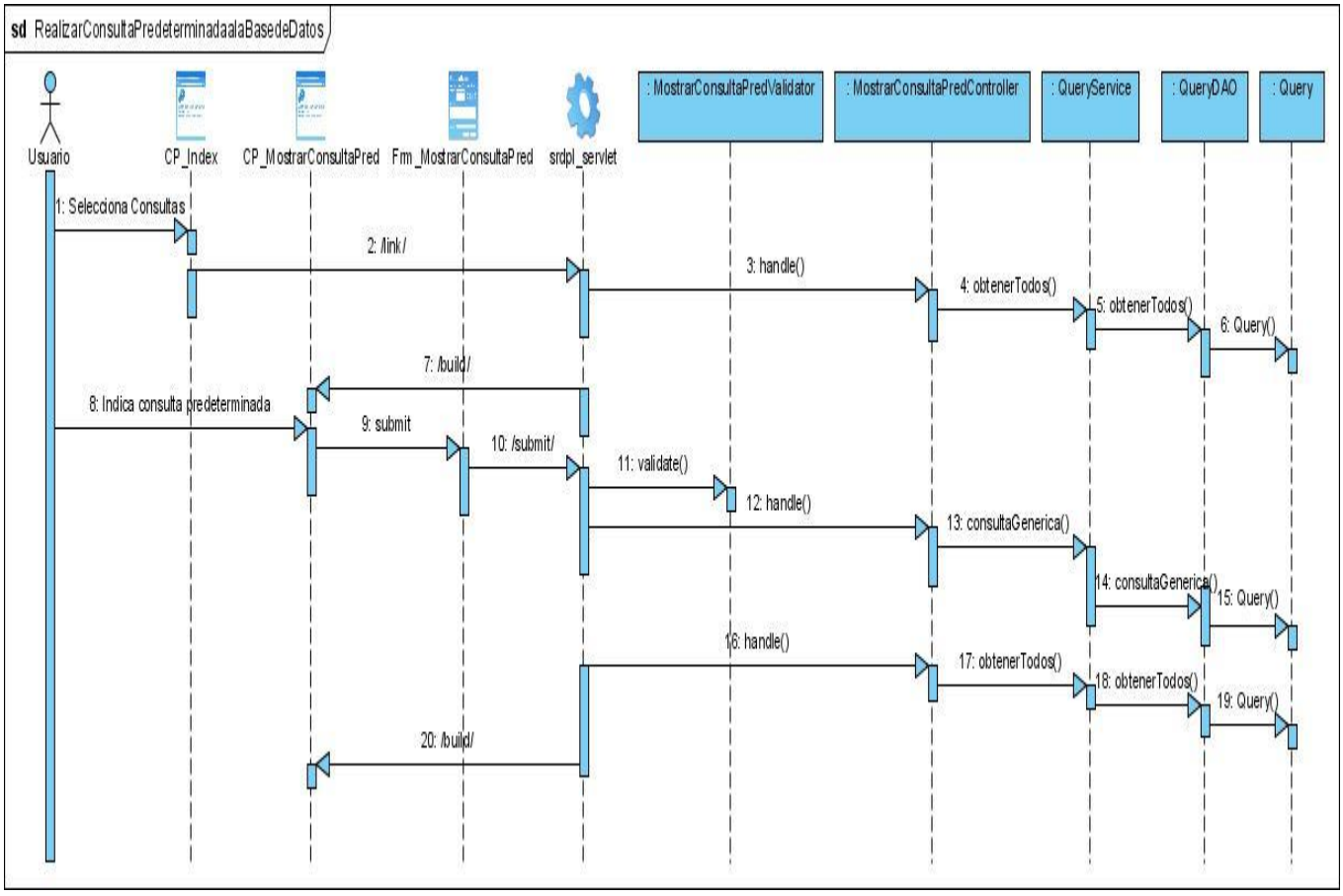


Fig. 34 DSCU Consultar información de la Base de Datos (Escenario Realizar Consulta Predeterminada a la Base de Datos)

- ✓ Escenario Diseñar Nueva Consulta a la Base de Datos.

En caso de que la consulta que el usuario desea realizar no se encuentre entre las predeterminadas que muestra el sistema, el usuario puede diseñar una nueva consulta, la Fig.35 muestra las interacciones de las clases que son necesarias para que se realice la petición del usuario.

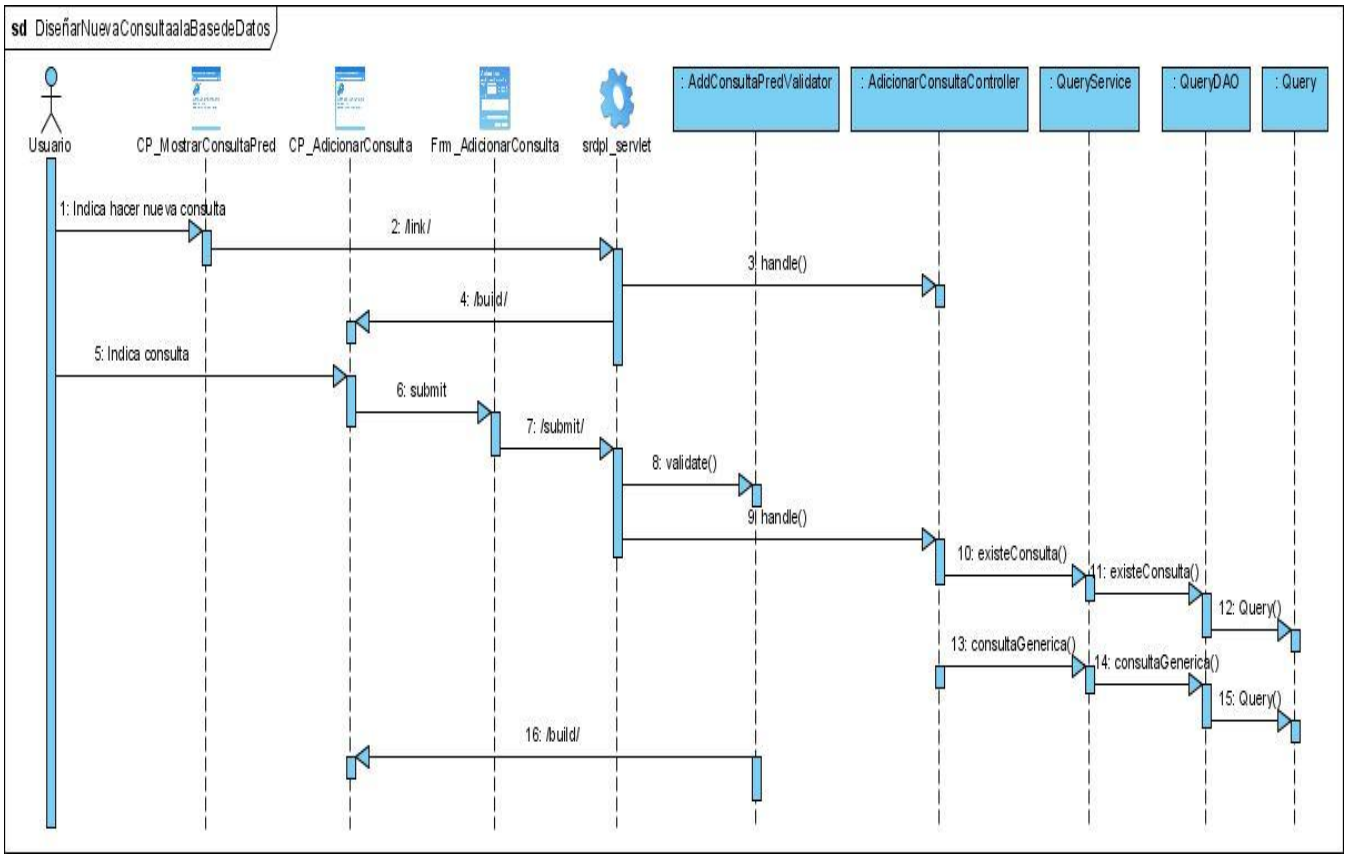


Fig. 35 DSCU Consultar información de la Base de Datos (Escenario Diseñar Nueva Consulta a la Base de Datos)

- ✓ Escenario Consultar Vista Lógica de la Base de Datos.

Para entender mejor la distribución de las tablas de la base de datos cuando el usuario va a diseñar una consulta, puede seleccionar la opción “Diseño de clases de la base de datos”. La Fig.36 muestra las interacciones de las clases necesarias que son necesarias para que se realice la petición del usuario.

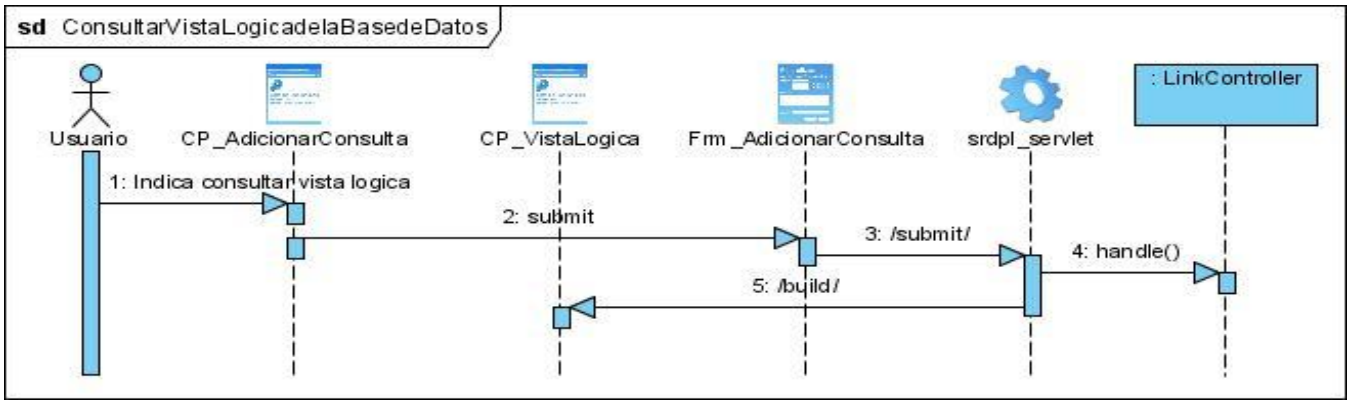


Fig. 36 DSCU Consultar información de la Base de Datos (Escenario Consultar Vista Lógica de la Base de Datos)

- ✓ Escenario Consultar Fichero PDB.

El diseño del sistema permite al usuario consultar la información de un fichero PDB desde cualquier página del. La Fig.37 muestra las interacciones que se producen entre las clases que son necesarias para que se realice la petición del usuario. Las mismas interacciones entre clases se producirían si el usuario hace la petición desde otra página cliente, cambiando en el diagrama de interacción sólo la página desde la cual se inicia la petición.

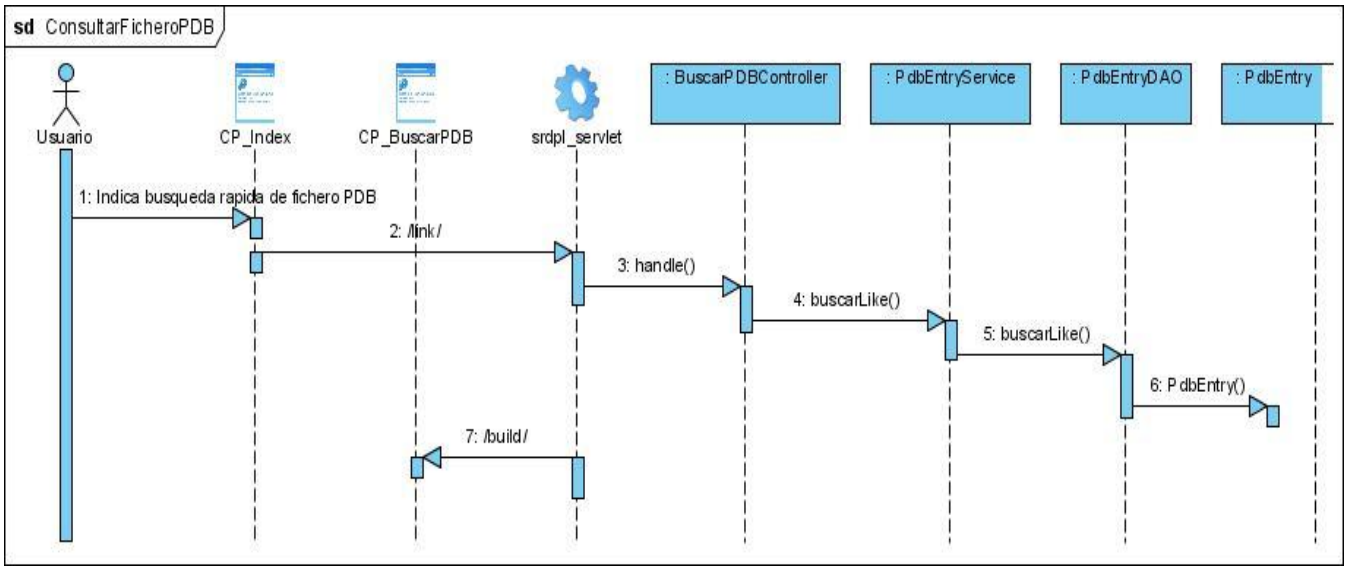


Fig. 37 DSCU Consultar información de la Base de Datos (Escenario Consultar Fichero PDB)

- ✓ Escenario Consultar Información de Fichero PDB.

El diseño del sistema permite que el usuario pueda consultar información de un fichero PDB desde tres páginas clientes distintas: CP_MostrarConsultaPred, CP_BuscarPDB y CP_AdicionarConsulta. La petición solo podrá hacerse después de realizada una consulta (Ver Fig.34, Fig.35 y Fig.37) el acceso a un fichero PDB mostrará el listado de los ligandos y sitios de enlace relacionados con la consulta realizada anteriormente. Las interacciones entre las clases que son necesarias para que se realice la petición del usuario son las mismas, solo cambiarían las páginas de inicio de la solicitud.

La Fig.38 muestra las interacciones entre las clases que son necesarias para que se realice la petición del usuario iniciando en la CP_ BuscarPDB.

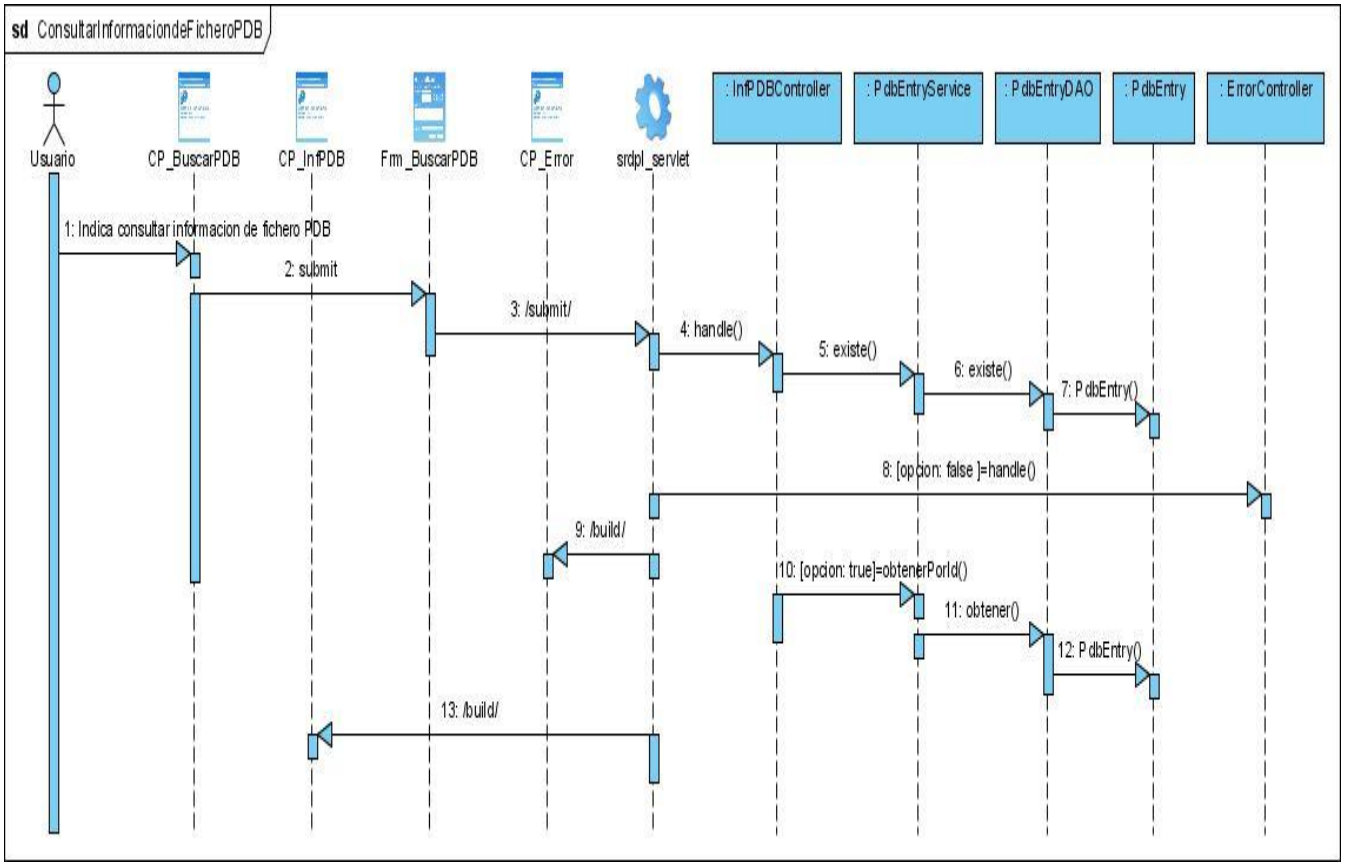


Fig. 38 DSCU Consultar información de la Base de Datos (Escenario Consultar Información de Fichero PDB)

- ✓ Escenario Consultar Información de Ligando.

La petición de “Consultar Información de Ligando” sólo podrá hacerse después de “Consultar Información de fichero PDB” (Ver Fig. 38). La Fig.39 muestra las interacciones de las clases que son necesarias para que se realice la petición del usuario.

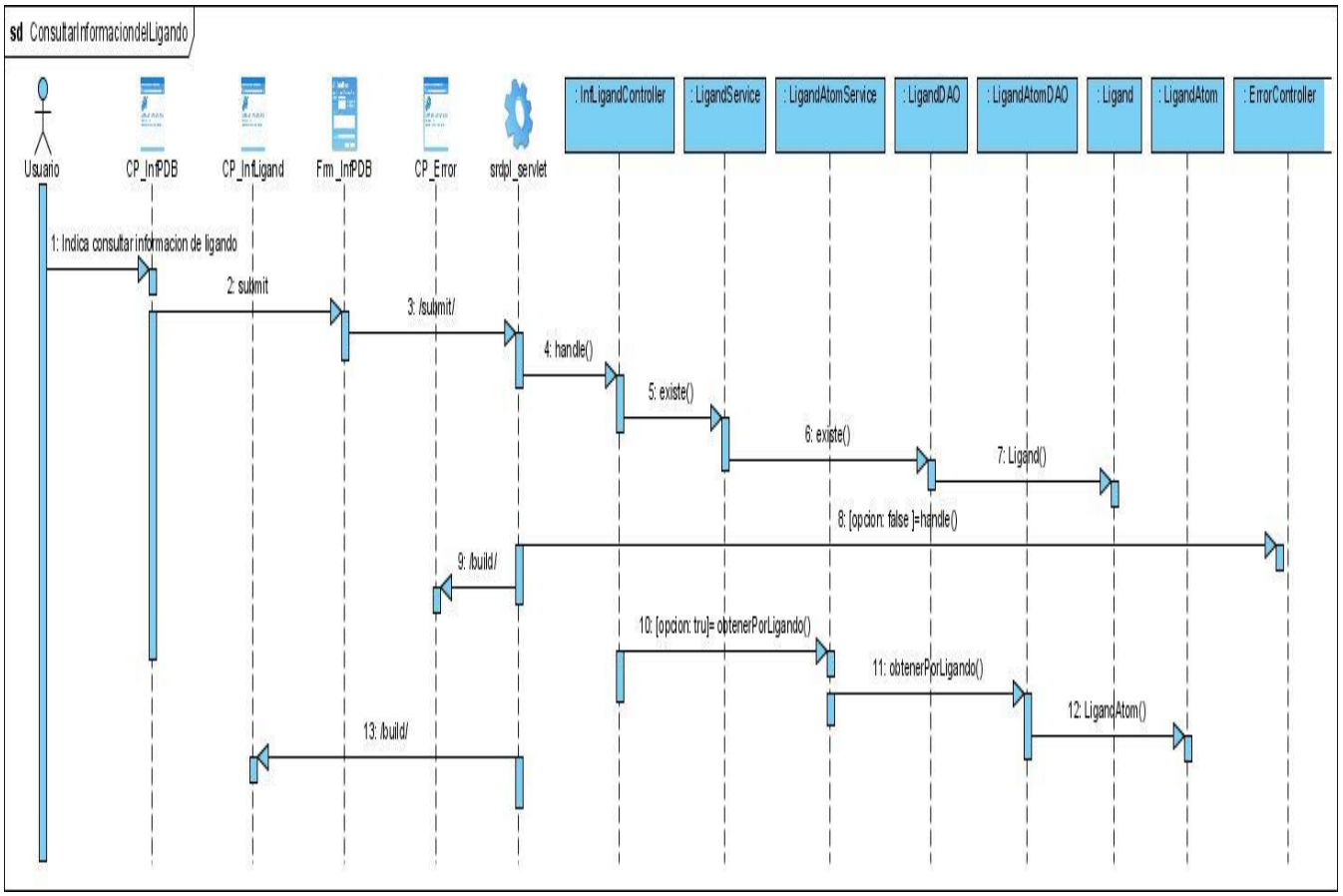


Fig. 39 DSCU Consultar información de la Base de Datos (Escenario Consultar Información de Ligando)

- ✓ Escenario Consultar Información de Sitio de Enlace.

La petición de Consultar Información de Sitio de Enlace es similar al Escenario Consultar Información de Ligando, sólo podrá hacerse después de Consultar la Información de un fichero PDB (Ver Fig. 38). La Fig.37 muestra las interacciones de las clases que son necesarias para que se realice la petición del usuario.

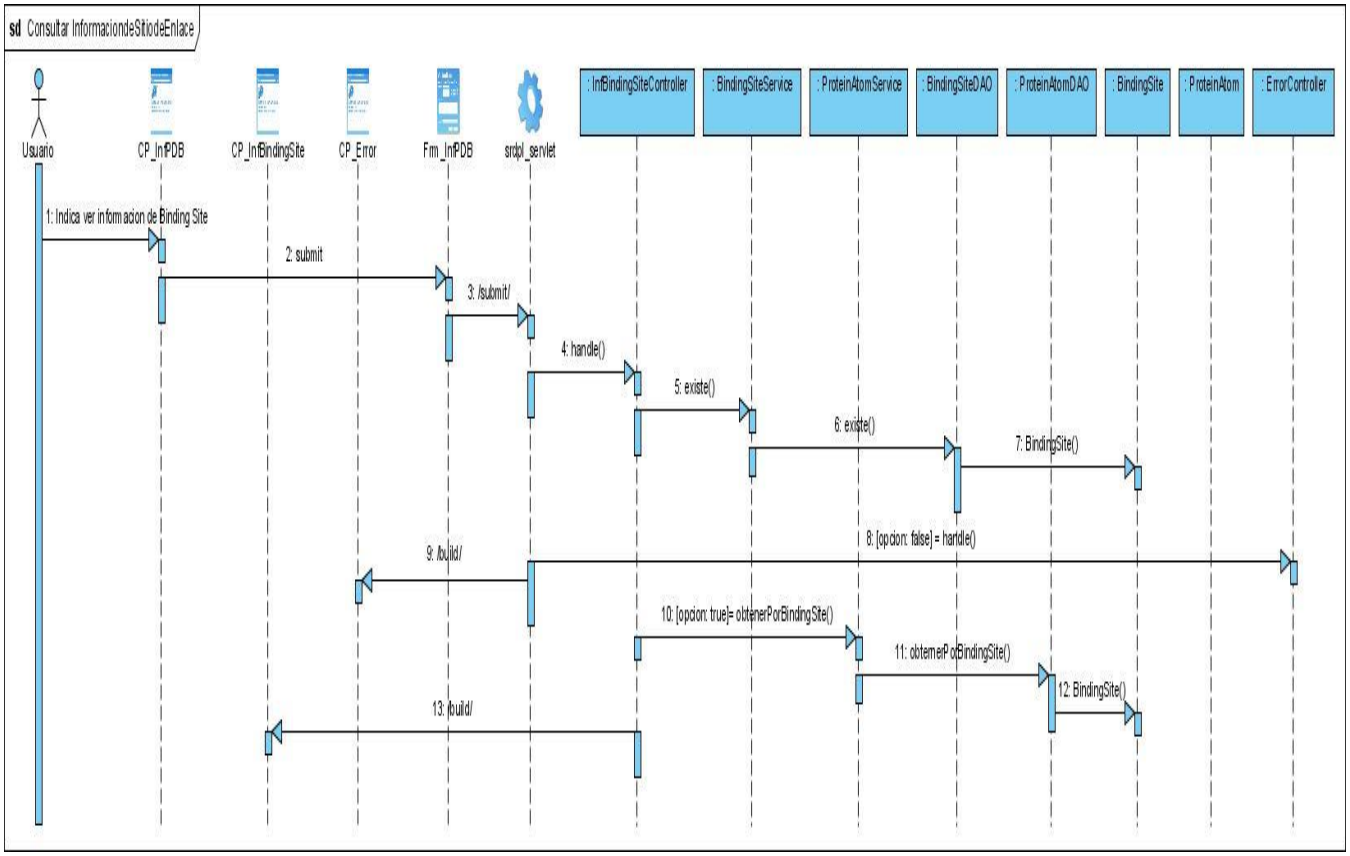


Fig. 40 DSCU Consultar información de la Base de Datos (Escenario Consultar Información de Sitio de Enlace)

Caso de Uso Gestionar Consultas Predeterminadas.

- ✓ Escenario Insertar Consulta Predeterminada.

La petición de insertar una consulta sólo puede ser realizada por el administrador después de diseñada una nueva consulta (Ver Fig.35). La Fig.41 muestra las interacciones de las clases necesarias para que se realice la petición del administrador.

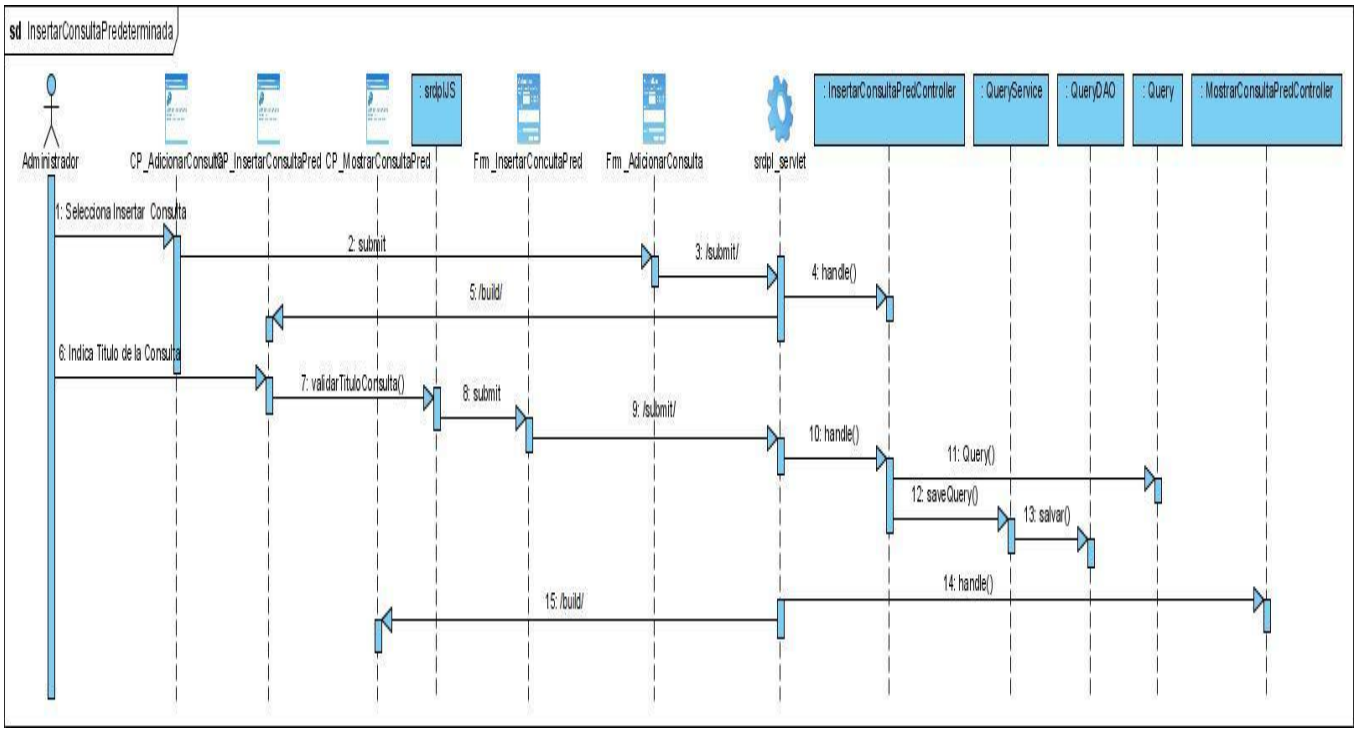


Fig. 41 DSCU Gestionar Consultas Predeterminadas (Escenario Insertar Consulta Predeterminada)

- ✓ Escenario Listar Consultas Predeterminadas.

En la sección “Administración” el administrador puede seleccionar la opción “Listar Consultas Predeterminadas” y puede indicar eliminar una consulta predeterminada. La Fig.42 muestra las interacciones de las clases que son necesarias para que se realice la petición del administrador.

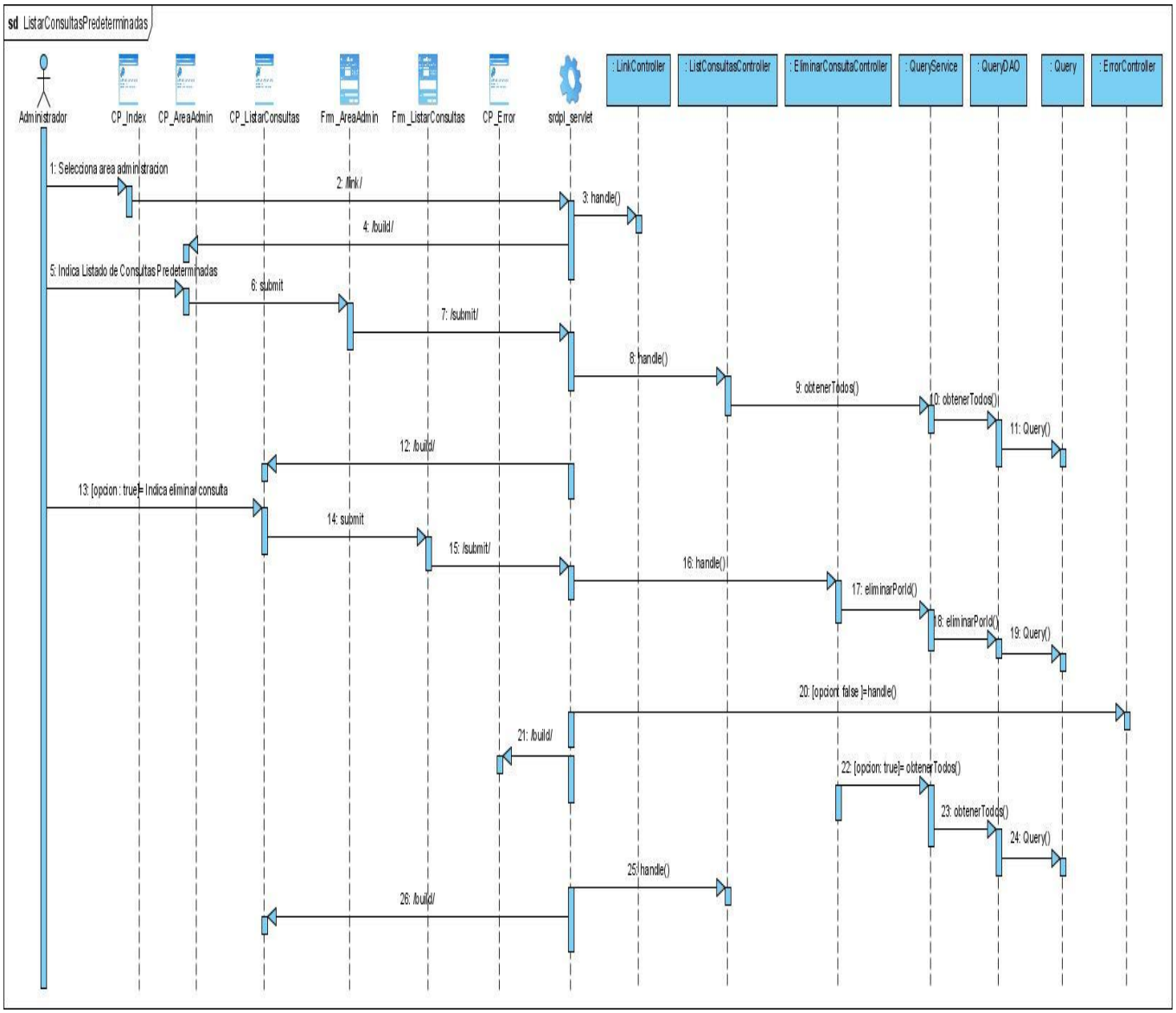


Fig. 42 DSCU Gestionar Consultas Predeterminadas (Escenario Listar Consultas Predeterminadas)

Caso de Uso Exportar Resultado de Consulta (Caso de Uso Extendido).

- ✓ Escenario Exportar Consulta en Formato .tar.gz.

Para exportar una consulta en formato .tar.gz es necesario realizarla previamente (Ver Fig.34 y Fig.35). La aplicación está diseñada para que el usuario pueda exportar una consulta realizada desde dos páginas clientes distintas: CP_MostrarConsultaPred y CP_AdicionarConsulta. Las interacciones entre

las clases que son necesarias para que se realice la petición del usuario son las mismas, sólo cambiarían las páginas clientes donde pudiera iniciarse la misma.

La Fig.43 muestra las interacciones entre las clases que son necesarias para que se realice la petición del usuario iniciando en la CP_ AdicionarConsulta.

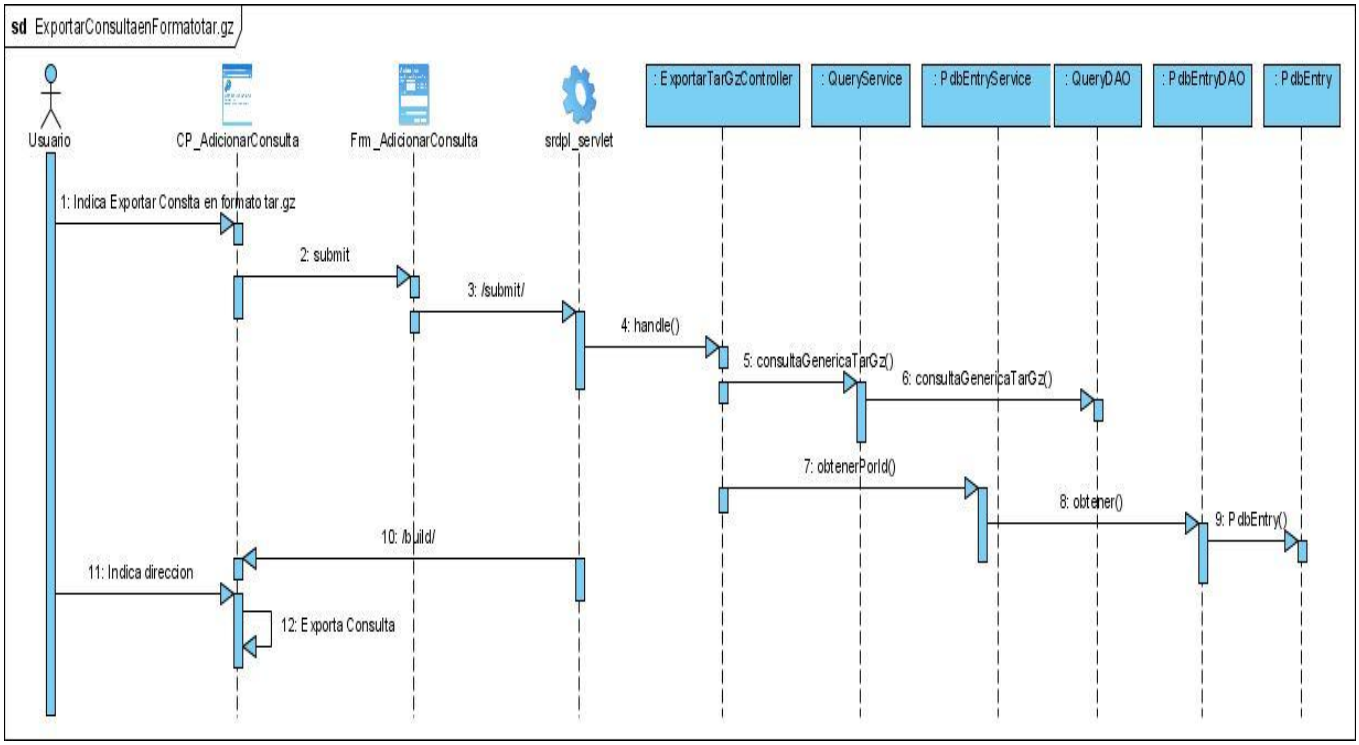


Fig. 43 DSCU Exportar Resultado de Consulta (Escenario Exportar Consulta en Formato .tar.gz)

- ✓ Escenario Exportar Consulta en Formato ligand.pdb.

Para exportar una consulta en formato ligand.pdb es necesario hacer una consulta previamente y ver la información de un fichero PDB relacionado con la misma (Ver Fig. 38). La Fig.44 muestra las interacciones de las clases que son necesarias para que se realice la petición del usuario.

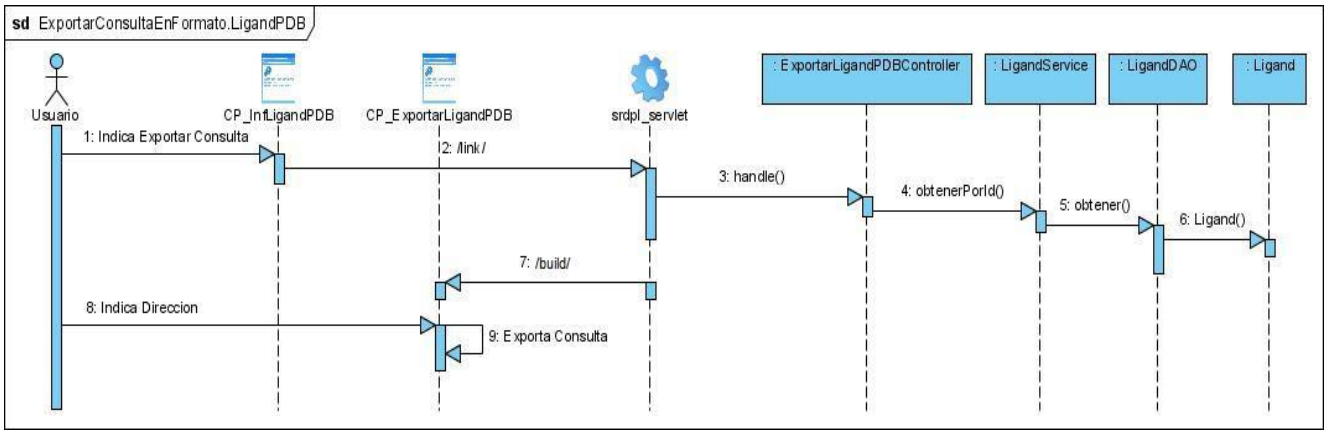


Fig. 44 DSCU Exportar Consulta (Escenario Exportar Consulta en Formato ligand.pdb)

- ✓ Escenario Exportar Consulta en Formato .contact.table.

Exportar una consulta en formato .contact.table es similar a exportarla en ligand.pdb, es necesario hacer una consulta previamente y ver la información de un fichero PDB relacionado con la misma (Ver Fig. 38). La Fig. 45 muestra las interacciones de las clases que son necesarias para que se realice la petición del usuario.

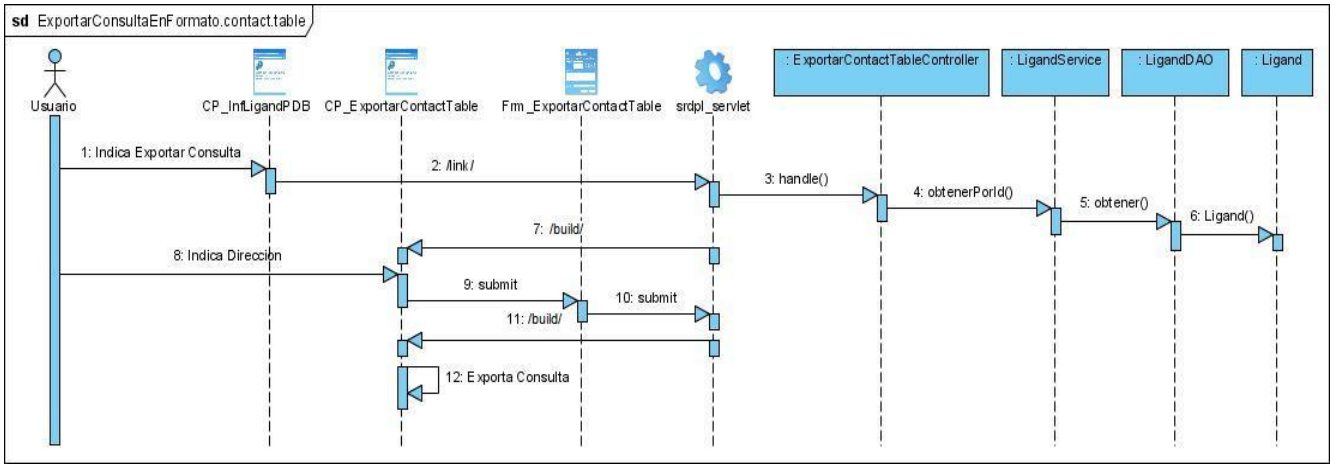


Fig. 45 DSCU Exportar Consulta (Escenario Exportar Consulta en Formato .contact.table)

3.3 REDISEÑO DE LA BASE DE DATOS

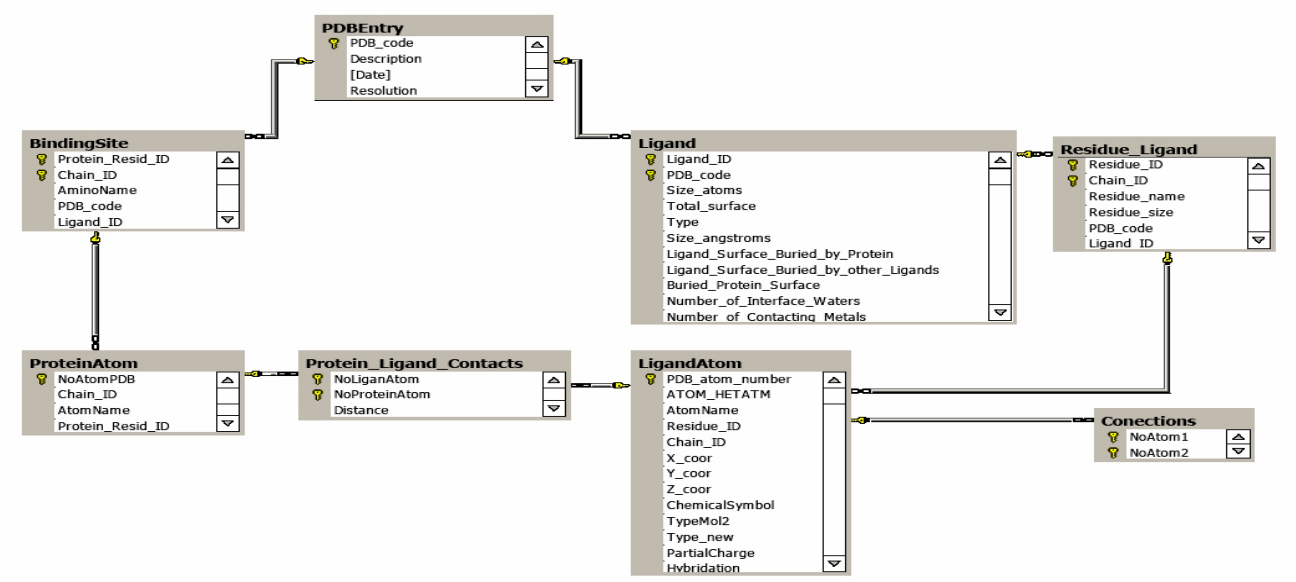


Fig. 46 Diseño inicial de la base de datos

La figura anterior muestra el rediseño que se le realizó a la base de datos inicial. La misma se ha rediseñado con el objetivo de dar solución a los nuevos requerimientos de los usuarios del CIM. Por ejemplo, se han agregado nuevos campos de interés en algunas tablas, tales como Ligand, ligand_pdb_text campo en el que se almacena información acerca de la distribución en el espacio de los átomos del ligando; además se han adicionado los campos contact_table_text y mor_text. En la tabla LigandAtom se han eliminado los campos type_new, partial_charge y hybridation y se han agregado los campos occupation , betafactor y type_cim, este último que determina una clasificación para cada átomo creada por los investigadores del CIM. De la misma manera se han añadido nuevas tablas con el propósito de tener un control de los usuarios que acceden al sistema y los comentarios que pueda presentar cada uno de ellos.

Con la nueva base de datos proteína – ligando del CIM se ha logrado un diseño orientado a consultas que simplifica en gran medida el texto de las consultas debido a que para los investigadores es muy importante tener presente en cada momento el complejo proteína - ligando al cual se hace referencia, o sea, su código PDB.

Como se puede observar se tiene conocimiento en cada tabla de que entrada PDB proviene, evitándose de esta manera enlaces a través de inner joins entre las tablas y el consecuente costo en el rendimiento debido a la gran cantidad de tuplas que posee la base de datos.

Se utilizará HQL como lenguaje para el manejo de consultas a la base de datos. Este lenguaje es similar a SQL y es utilizado para obtener objetos de la base de datos según las condiciones

especificadas en el HQL. El uso de HQL permite usar un lenguaje intermedio, que según la base de datos que se use, será traducido al SQL dependiente de cada base de datos de forma automática y transparente; además es un lenguaje de consultas orientado a objetos, descriptivo y fácil de aprender. Por ejemplo si se quisiera recuperar de la base de datos todas las entradas proteína – ligando se escribiría de la siuiente forma:

SQL

```
select id_pdb_code, resolution, date_entry, description from pdb_entry
ó
select * from pdb_entry
```

HQL

```
from PdbEntry
```

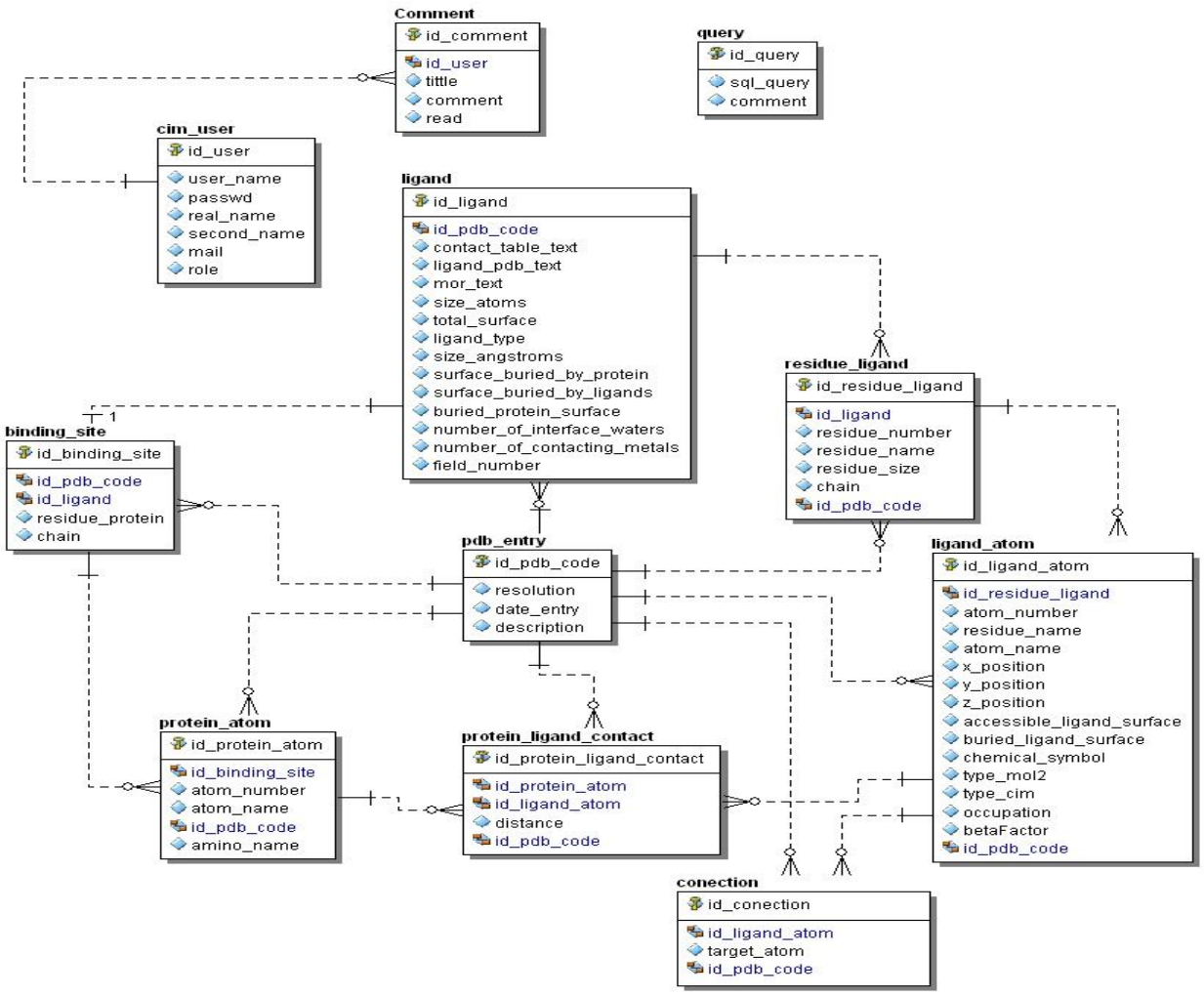


Fig. 47 Vista Lógica de Base de Datos Actual

3.4 DIAGRAMA DE DESPLIEGUE

El diagrama de despliegue muestra la distribución física sobre la que será desplegado el software. Presenta los nodos computacionales que intervienen en el funcionamiento del sistema, las conexiones entre estos y los protocolos de comunicación.

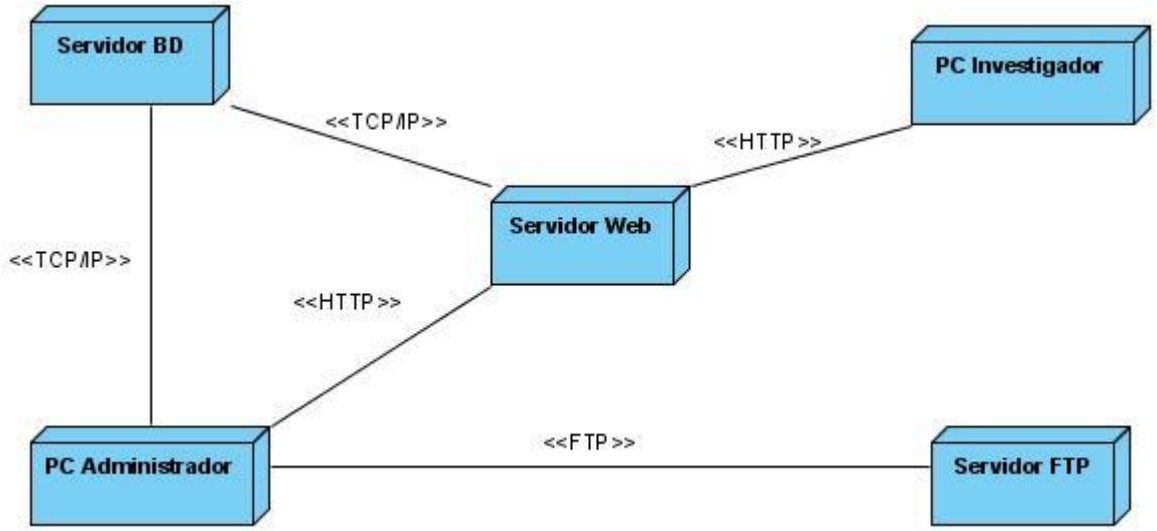


Fig. 48 Diagrama de Despliegue

En la PC del administrador se puede descargar la aplicación escritorio desde el servidor de la aplicación Web y a través de la aplicación escritorio puede descargar los archivos PDB que se encuentran en el Servidor FTP, procesarlos y almacenar el resultado directamente en el Servidor de Base de Datos. La PC del Investigador tendrá acceso solamente al servidor donde se encuentra la aplicación Web.

CONCLUSIONES

En este capítulo se mostraron los diagramas de clase de diseño y secuencia generados durante el flujo de trabajo de Análisis y Diseño para las aplicaciones Web y escritorio. Se presentó además la vista lógica de la base de dados actual que cumple con los requerimientos del CIM y el diagrama de despliegue para mostrar la distribución física del sistema.

CAPÍTULO 4: IMPLEMENTACIÓN DEL SISTEMA

INTRODUCCIÓN

En este capítulo se hará una descripción de cómo se implementarán los elementos del Modelo de Diseño en termino de componentes. Se mostrará la vista lógica de implementación del sistema y los diagramas de componente generados para cada caso de uso. Se describirán los algoritmos más importantes del sistema y se mostrarán algunas imágenes del mismo.

4.1 VISTA DE IMPLEMENTACIÓN DEL SISTEMA

En el Flujo de Trabajo de Implementación se empieza con el resultado del diseño y se implementa el sistema en términos de componentes. Un componente es una unidad física de implementación que encapsula una o más clases del diseño. En resumen, los diagramas de componentes muestran los elementos físicos reales más significativos del sistema.

La vista de implementación del sistema queda constituida por un grupo de componentes y subsistemas de implementación que modelan el empaquetado real del sistema. Estos subsistemas de implementación representan la estructura de carpetas definidas para el proyecto, y contienen los ficheros modelados en término de componentes. A continuación se muestra la vista de los principales elementos de los diagramas de componentes.

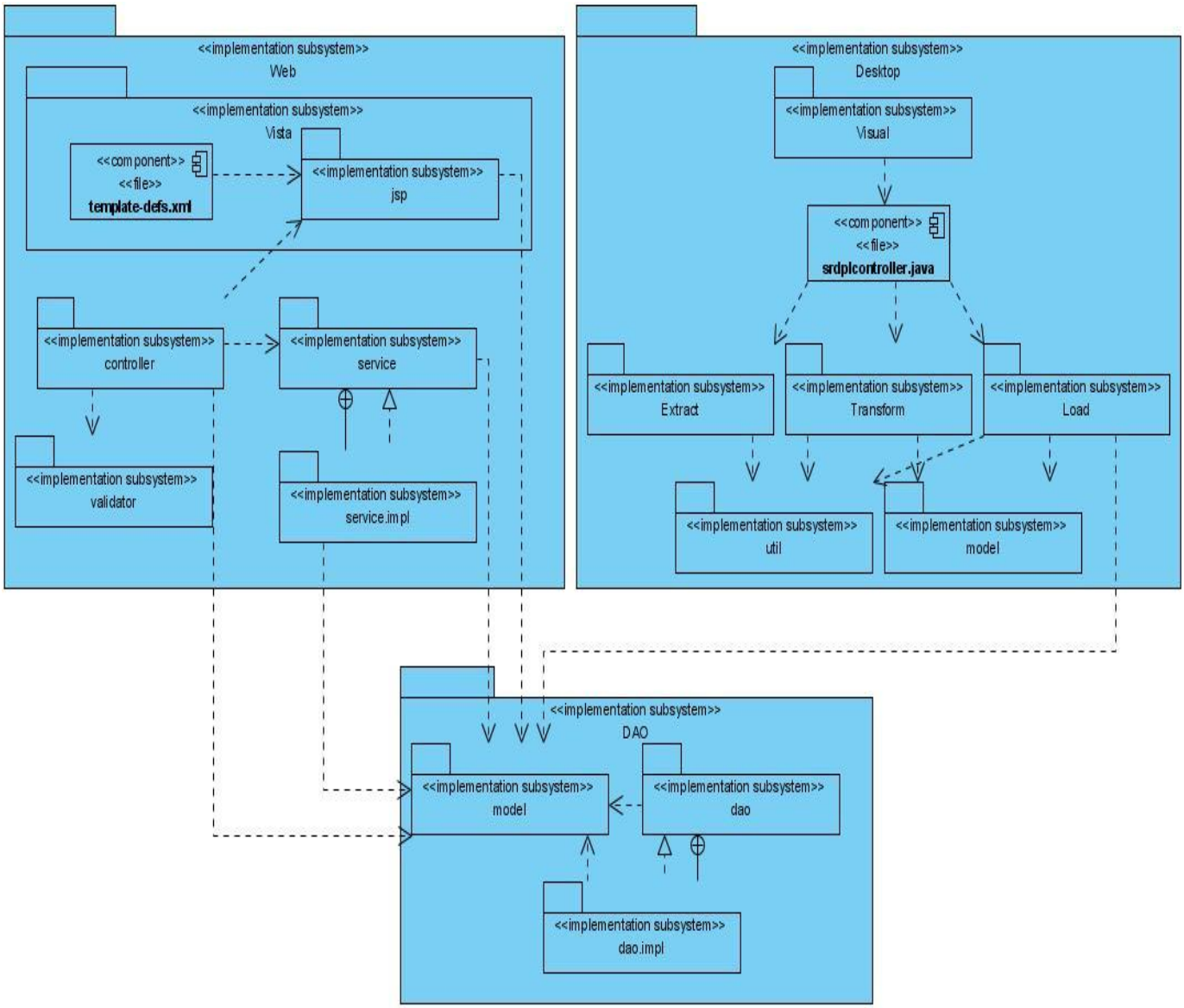


Fig. 49 Vista de Implementación

A continuación se expone la descripción de los componentes y subsistemas de implementación que conforman la vista de implementación.

Subsistema de implementación Web: encapsula los componentes de la aplicación Web.

Subsistema de implementación Vista: encapsula los componentes que brindan la interfaz gráfica de usuario de la aplicación Web.

Subsistema de implementación jsp: agrupa los componentes que implementan el código correspondiente a cada *Java Server Pages.*

Componente template-defs.xml: componente que implementa la clase Layout del diseño. Contiene los elementos que definen la estructura de las páginas.
Subsistema de implementación controller: encapsula los componentes que se especializan en atender las peticiones del cliente.
Subsistema de implementación validator: agrupa los componentes que validan la entrada de datos a la aplicación.
Subsistema de implementación service: encapsula los componentes que definen las interfaces de las reglas del negocio.
Subsistema de implementación service.impl: encapsula los componentes que implementan las interfaces de los servicios.
Subsistema de implementación Desktop: encapsula los componentes de la aplicación escritorio.
Subsistema de implementación Visual: encapsula los componentes que brindan la interfaz gráfica de usuario de la aplicación escritorio.
Componente srdplcontroller: componente que implementa la solicitud de los servicios.
Subsistema de implementación Extract: contiene los componentes que implementan el proceso de extracción de los datos.
Subsistema de implementación Transform: encapsula los componentes que implementan el proceso de transformación de los datos.
Subsistema de implementación Load: contiene los componentes que implementan el proceso de carga de los datos.
Subsistema de implementación útil: contiene los componentes que implementan los buffer intermedios entre los procesos de extracción, transformación y carga de datos. Además contiene el componente PDBMor.java que representa una clase entidad útil que se utiliza la aplicación escritorio para almacenar en memoria las clases entidades que serán cargadas en la base de datos.
Subsistema de implementación model (Desktop): agrupa los componentes que implementan las clases entidades necesarias para almacenar la información que se obtiene de los ficheros que son el resultado de la transformación de los archivos PDB.
Subsistema de implementación DAO: encapsula los componentes que representan las entidades del negocio y los dao.
Subsistema de implementación dao: agrupa las interfaces para los dao que dan persistencia a determinada entidad.
Subsistema de implementación dao.impl: encapsula los componentes que implementan las interfaces de los dao.

Subsistema de implementación modelo: contiene los componentes que representan los datos o entidades de negocio.

4.1.1 Diagrama de componentes de la aplicación escritorio

La siguiente figura muestra en detalles el diagrama de componentes para el módulo escritorio. El subsistema de implementación Extract muestra los componentes necesarios para el proceso de extracción de datos; los componentes que representan la interfaz IDowloader y su implementación solamente serán solicitados por el servicio IExtractService en caso de que la solicitud del cliente sea almacenar ficheros desde el servidor FTP. En el subsistema de implementación Transform el componente complex_info es el algoritmo implementado por los investigadores del CIM que transforma los ficheros PDB. En el subsistema de implementación DAO se muestran sólo las interfaces de los dao que son necesarios en la aplicación escritorio.

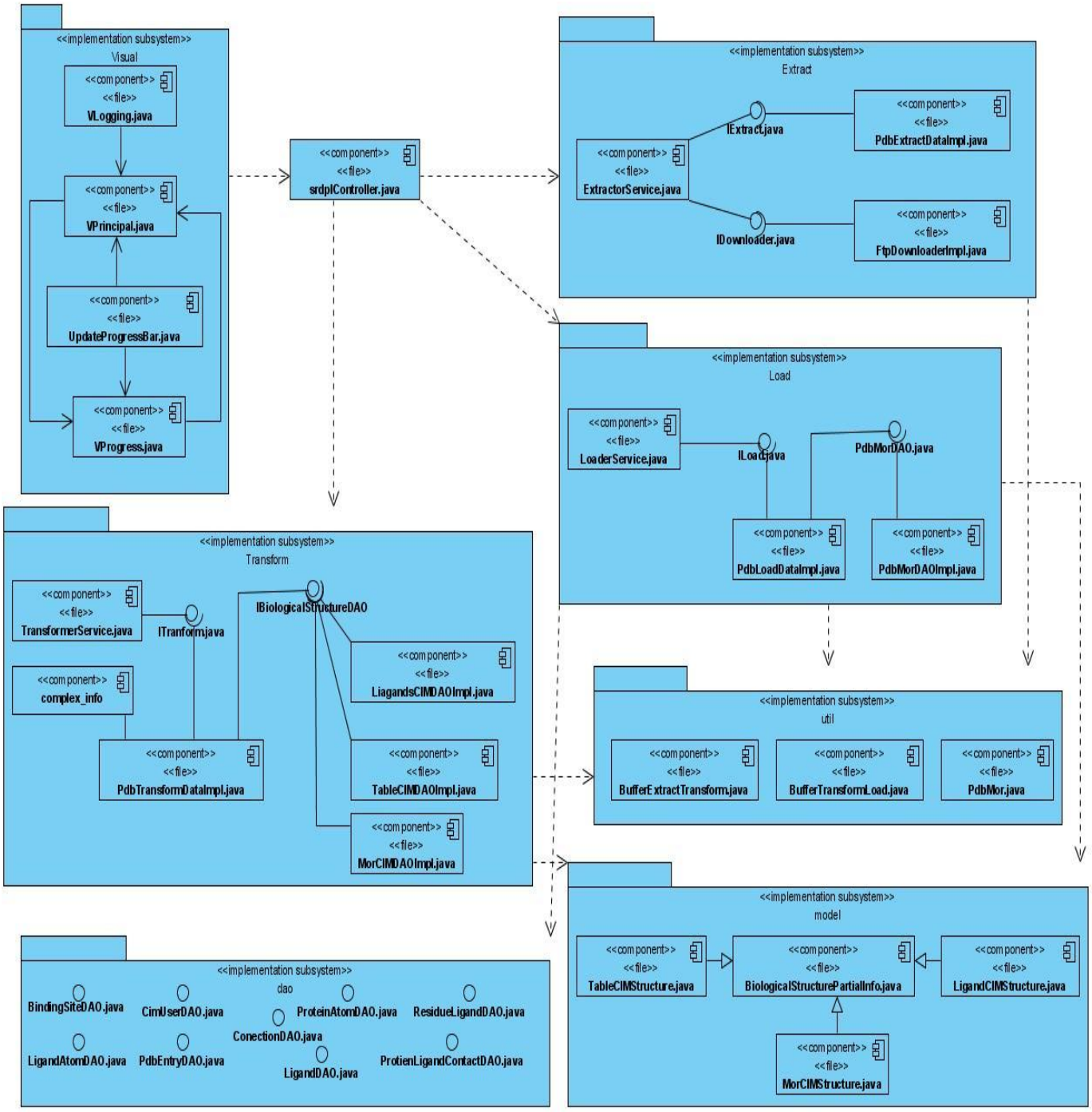


Fig. 50 Diagrama de componentes aplicación escritorio

4.1.2 Diagramas de componentes de la aplicación Web

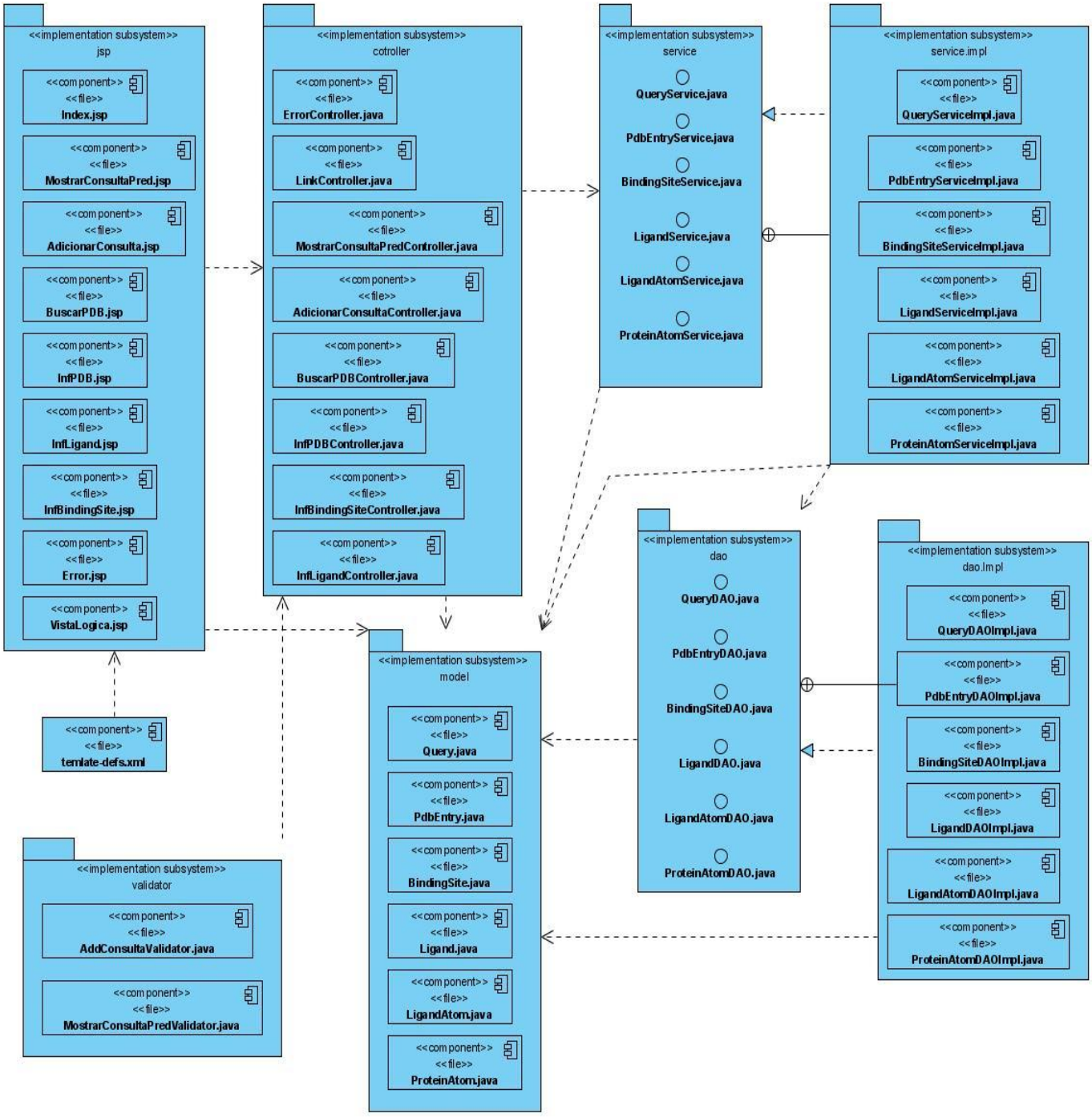


Fig. 51 Diagrama de componentes Caso de Uso Consultar Información de la Base de Datos

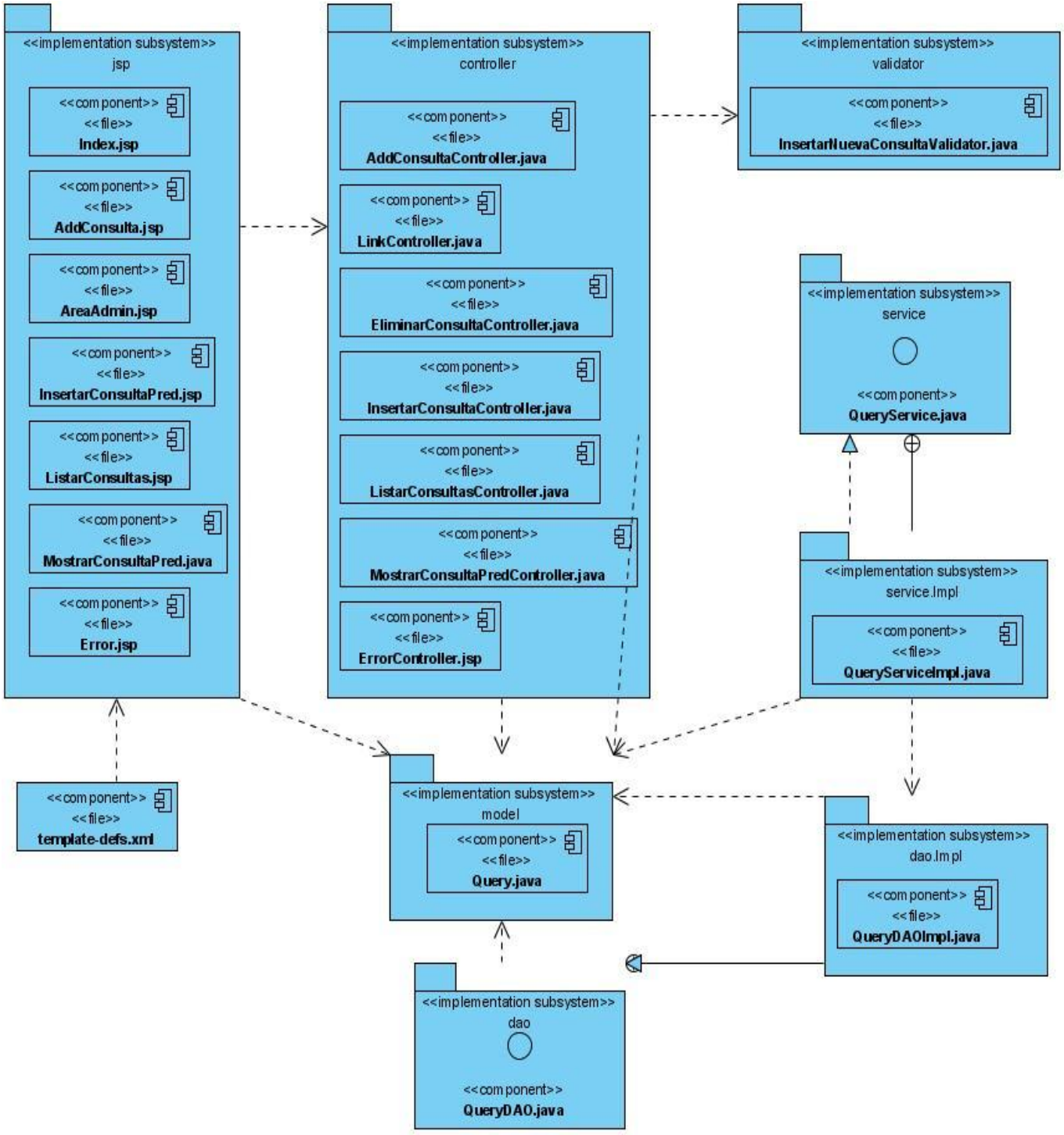


Fig. 52 Diagrama de componentes Caso de Uso Gestionar Consultas Predeterminadas

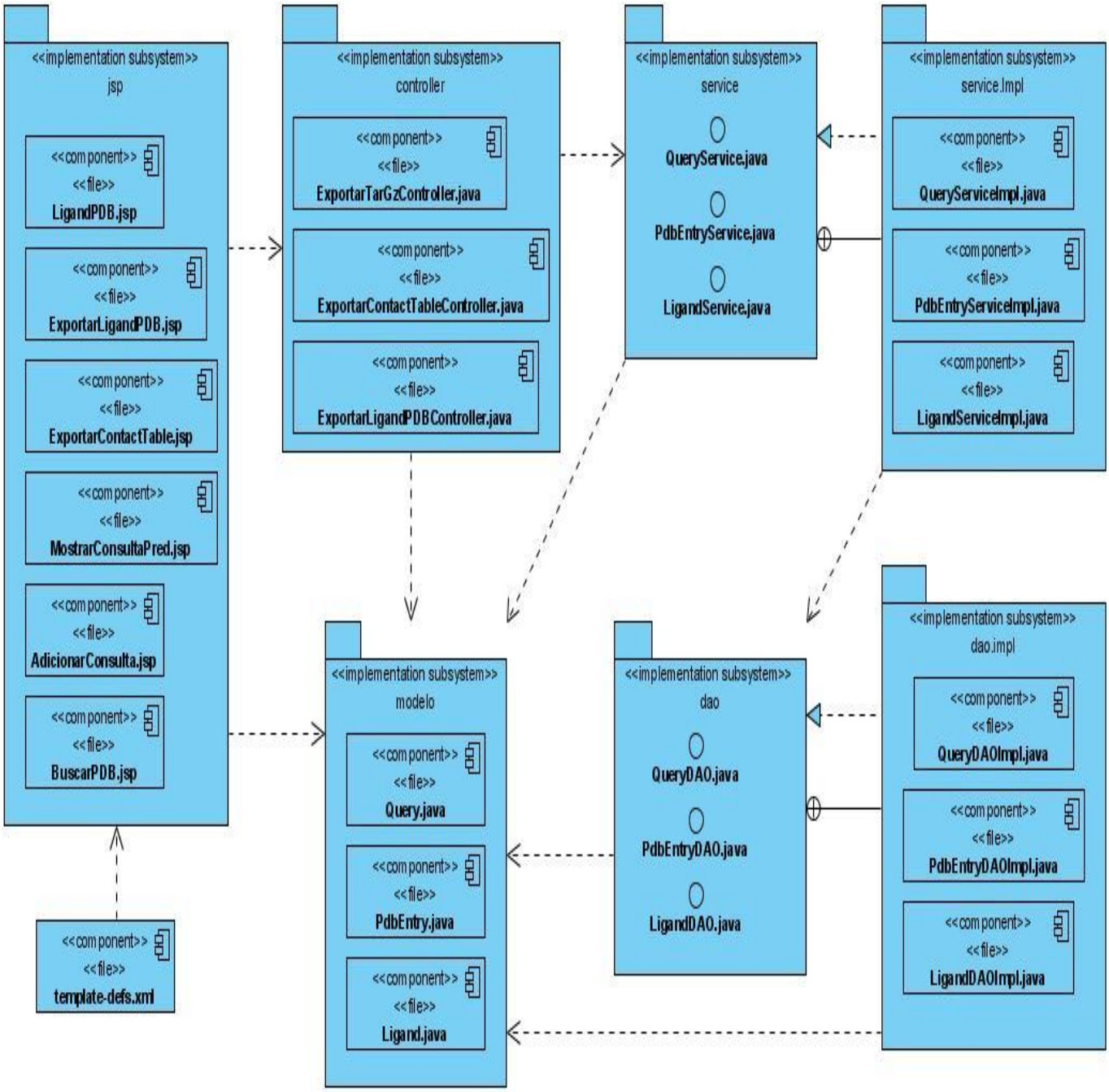


Fig. 53 Diagrama de componentes Caso de Uso Exportar Resultado de Consulta

4.3 ALGORITMOS MÁS IMPORTANTES DE LA APLICACIÓN ESCRITORIO

El método void run() que se presenta pertenece a la clase ExtractorService. Es el encargado de realizar el proceso de extracción a los ficheros PDB. La entrada es la dirección de los ficheros PDB y la salida el buffer intermedio de los procesos de extracción y transformación, con las direcciones de los ficheros. Primeramente se comprueba si la dirección de los ficheros proviene de un servidor ftp, de ser así se descargan los ficheros desde el servidor en un arreglo y se les aplican las reglas de extracción. Luego se almacena el resultado en el buffer. Si la dirección de los ficheros proviene de un directorio local entonces se construye una lista con las direcciones locales de los ficheros. Se aplican las reglas de extracción a los ficheros y el resultado se almacena en el buffer. Finalmente se comprueba si se trata de un fichero, de ser así se le aplican las reglas de extracción y el resultado se almacena en el buffer.

```
D:= dirección de los ficheros
B:= buffer de extracción - transformación
si  D proviene de un servidor ftp entonces
        L:= descargar los ficheros del servidor ftp
        E:= L.primero.extraer
        para i:= 0 hasta L.tamaño hacer
                B <-L[i]
        fin para
fin si
sino si D proviene de un directorio local entonces
        L:=obtener la lista de las direcciones locales de los ficheros
        para i:= 0 hasta L.tamaño hacer
        E:= L.primero.extraer
                B <-L[i]
        fin para
fin sino si
sino
        F:= obtener la dirección del fichero
        E:= L.primero.extraer
        B <- F
fin sino
```

El siguiente método void run() pertenece a la clase TransformerService. Es el responsable de realizar el proceso de transformación a los ficheros PDB. La entrada es la información contenida en el buffer intermedio de los procesos extracción y transformación y la salida el buffer intermedio de los procesos de transformación y carga. Primeramente se obtienen los ficheros contenidos en el buffer intermedio

de los procesos de extracción y carga, se les aplican las reglas de transformación y el resultado se almacena en el buffer intermedio de los procesos de transformación y carga.

B:= buffer de extracción – transformación
S:= buffer de transformación - carga
mientras B tenga elementos **entonces**
    T:= B.primero.tranformar
    S <- T
fin mientras

El siguiente método void run() pertenece a la clase LoaderService. Es el responsable de realizar el proceso de carga del resultado de la transformación a la base de datos. Se obtiene el resultado del proceso de transformación contenido en el buffer intermedio de los procesos de transformación y carga y se almacena en la base de datos.

B:= buffer de transformación - carga
mientras B tenga elementos **entonces**
    S.cargar <- B.primero
fin mientras

4.4 ALGORITMOS MÁS IMPORTANTES DE LA APLICACIÓN WEB

A continuación se exponen fragmentos de código correspondientes a la implementación de la aplicación Web. Estos códigos corresponden al escenario Diseñar Nueva Consulta a la Base de Datos del Caso de Uso Consultar Información de la Base de Datos.

El controlador AdicionarConsultaController es el que procesa toda la información cuando se hace la petición de realizar una nueva consulta a la base de datos.

Cuando se da submit primero se ejecuta el método ModelAndView onSubmit (HttpServletRequest request, HttpServletResponse response, Object command, BindException errors) en el controlador. La entrada es el request de la petición y la salida la vista AdicionarConsultaPred.htm.

```
@Override
protected ModelAndView onSubmit(HttpServletRequest request,
        HttpServletResponse response, Object command, BindException errors)
        throws Exception {
    request.getSession().setAttribute("hqlQ",
            request.getParameter("sqlQuery"));
    return new ModelAndView(new RedirectView(getSuccessView()));
}
```

En el método lo que se hace es registrar en la sesión del usuario el texto de la consulta que ha enviado a ejecutarse, o sea:
R:= registra el texto de la consulta en la sesión del usuario
retornar AdicionarConsultaPred.htm

Luego se ejecuta el método Map referenceData (HttpServletRequest request, Object command, Errors errors). La entrada es el request de la petición del usuario y la salida los objetos a presentarse en la vista.
Primeramente se comprueba el texto de la consulta. Luego se verifica que exista la consulta en la base de datos y se obtiene el resultado de la ejecución de la consulta. Se comprueba el resultado de la ejecución de la consulta. Se verifica que la consulta realizada no sea sobre las tablas Query, CimUser o Comment. Luego se comprueba que en la consulta no aparezcan los operadores delete, update, create o select. Finalmente se envía el objeto a presentase en la vista.

```
S:= obtiene el texto de la consulta registrada en el request de la sesión del usuario
O:= objetos a mostrar en la vista
si S <> null entonces
        E:= verifica si existe la consulta en la base de datos
        L:= obtiene el resultado de la ejecución de la consulta
        si  L.tamaño > 0 cierto entonces
                si L[0] es de tipo Query, CimUser, Comment cierto entonces
                O[msg] <- “No se pueden efectuar consultas sobre estas tablas”
                fin si
                sino
                        si  S contiene delete, update, create, select entonces
                        O[msg] <- “Los operadores delete, update, create, select no están permitidos”
                        fin si
                fin sino
        fin si
fin si
O[list] <- L
O[existe] <- E
retornar O
```

4.5 INTEGRACIÓN DE LA APLICACIÓN ESCRITORIO CON LA APLICACIÓN WEB

Para la integración de la aplicación escritorio con la aplicación Web se utilizó la tecnología Java Web Start. Se creó una librería cimUtil.jar que contiene los ficheros y ejecutables del algoritmo

implementado por los investigadores del CIM, así como las clases para exportar estos a la PC cliente en una dirección determinada. De esta manera se garantiza que el procesamiento de los ficheros PDB se realice en dicha PC cliente. En el servidor Web mediante el fichero xml de configuración app.jnlp se logra crear la conexión entre la aplicación Web, la aplicación escritorio y sus dependencias.

```
<?xml version="1.0" encoding="UTF-8"?>
<jnlp spec="6.0+" codebase="http://10.34.19.205:5801/WINPL108/app-
desktop/" href="app.jnlp" >
    <information>
        <title>Procesar PDB</title>
            <vendor>Sistema Recopilacion Datos Proteina - Ligando
</vendor>
            <description>Envio de informacion a la BD</description>
            <description kind="short">APLICACION ESCRITORIO CIM
</description>
       <offline-allowed/>
    </information>

      <security>
          <all-permissions/>
      </security>

    <update check="always" policy="always" />

    <resources>
        <java version="1.6+"/>
        <jar href="srdpl.jar" main="true"/>
        <jar href="lib/acegi-security.jar"/>
            <!-- Aqui van las demas dependencias de la aplicacion
escritorio-->
        <jar href="lib/cimUtils.jar"/>
    </resources>

    <application-desc main-class="srdpl.visual.Main" />
</jnlp>
```

Fig. 54 Imagen JNLP

4.6 IMÁGENES DEL SISTEMA

A continuación se muestran imágenes del sistema. Se evidencia además en las imágenes que se presentan cómo quedó constituida la integración de la aplicación Web con la aplicación escritorio a través de la tecnología Java Web Start.

Fig. 55 Imagen Página Inicio

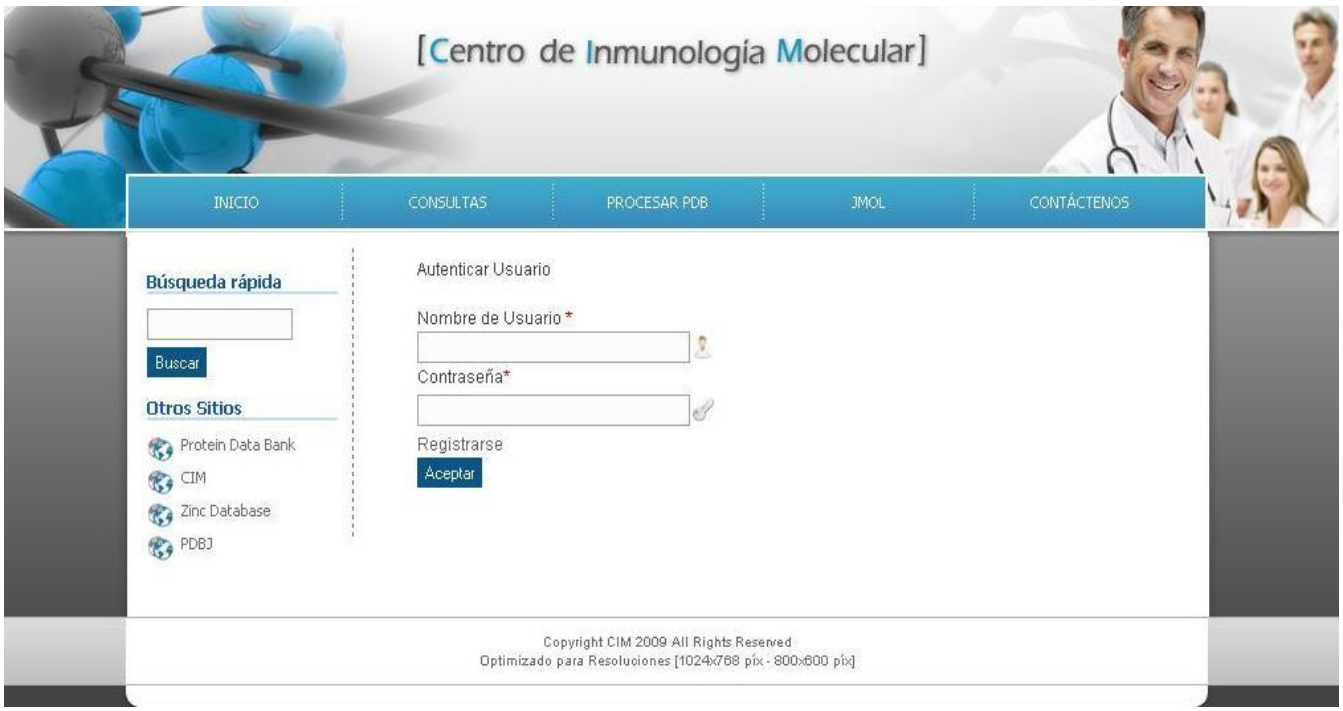


Fig. 56 Imagen Autenticar

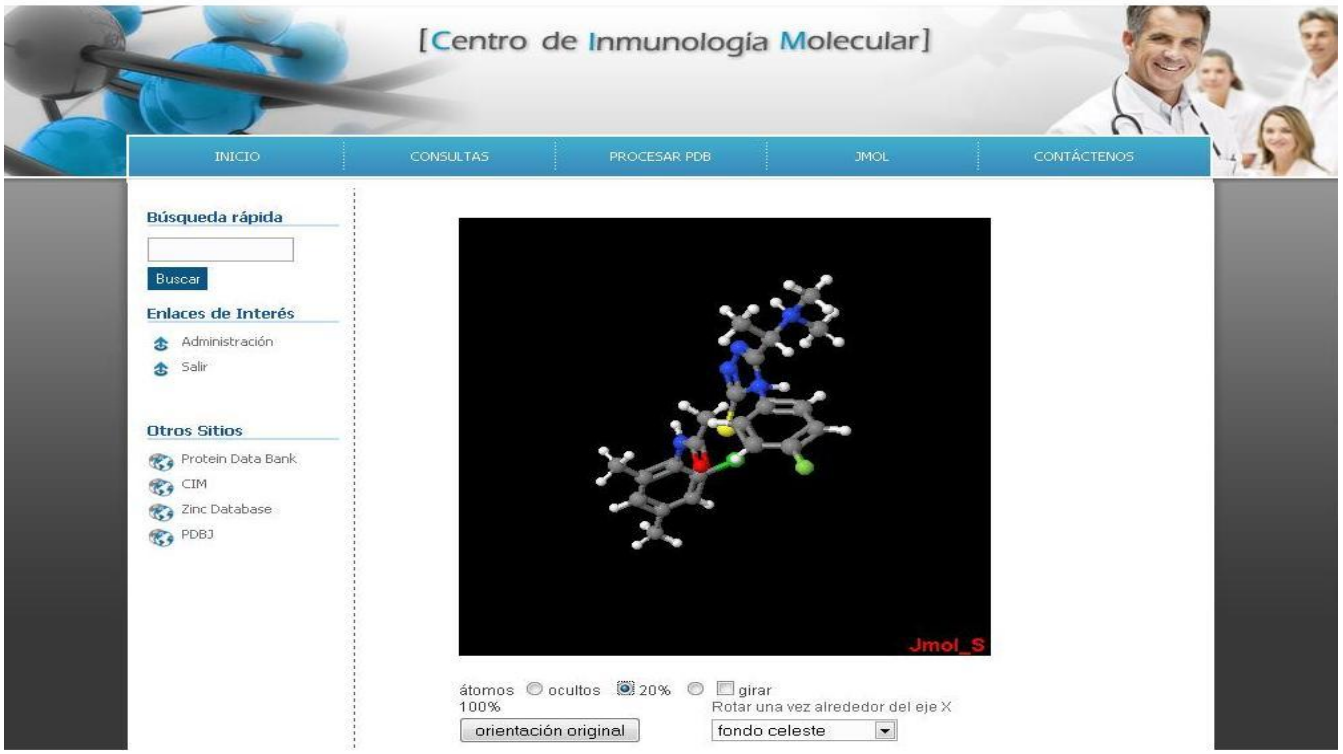


Fig. 57 Imagen Visualizar Proteína

PDB	idLigandAtom	AccessibleSurface	BuriedSurface	ChemicalSymbol	TypeMol2
1AIJ	6525	0.0	2.79	CSym	Mol2
	6526	0.0	7.47	CSym	Mol2
	6527	0.0	4.09	CSym	Mol2
	6528	0.0	5.01	CSym	Mol2
	6529	0.0	6.31	CSym	Mol2
	6530	0.0	3.38	CSym	Mol2
	6531	0.0	21.33	CSym	Mol2
	6532	0.0	27.94	CSym	Mol2
	6533	0.0	28.67	CSym	Mol2
	6534	2.71	12.21	CSym	Mol2
	6535	0.0	11.42	CSym	Mol2
	6536	0.0	3.58	CSym	Mol2
	6537	10.53	27.43	CSym	Mol2
	6538	2.08	27.15	CSym	Mol2
	6539	0.0	16.01	CSym	Mol2
	6540	0.0	6.85	CSym	Mol2

Fig. 58 Imagen Consulta Predeterminada

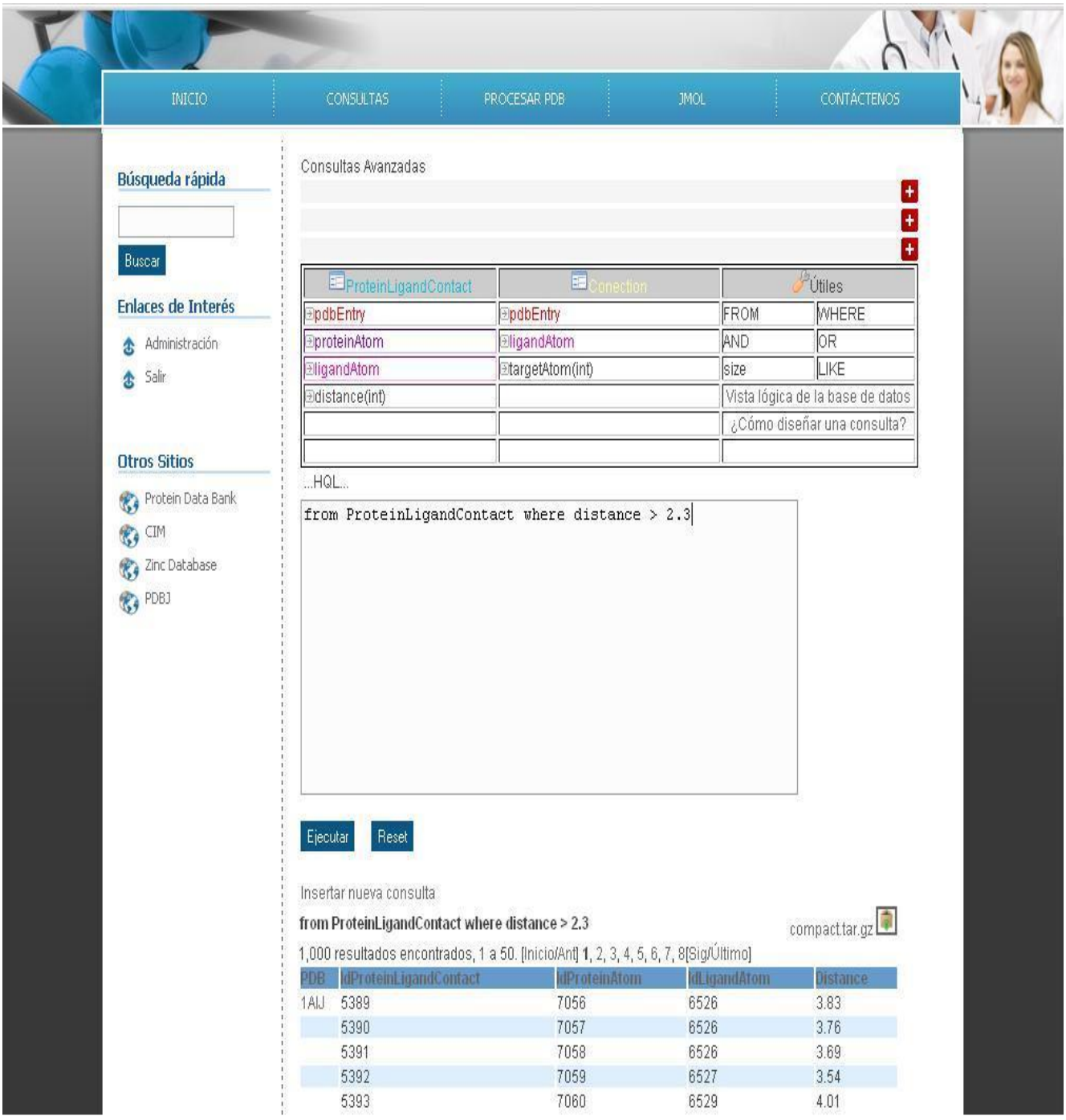


Fig. 59 Imagen Adicionar Consulta

Fig. 60 Imagen Iniciando Java Web Start

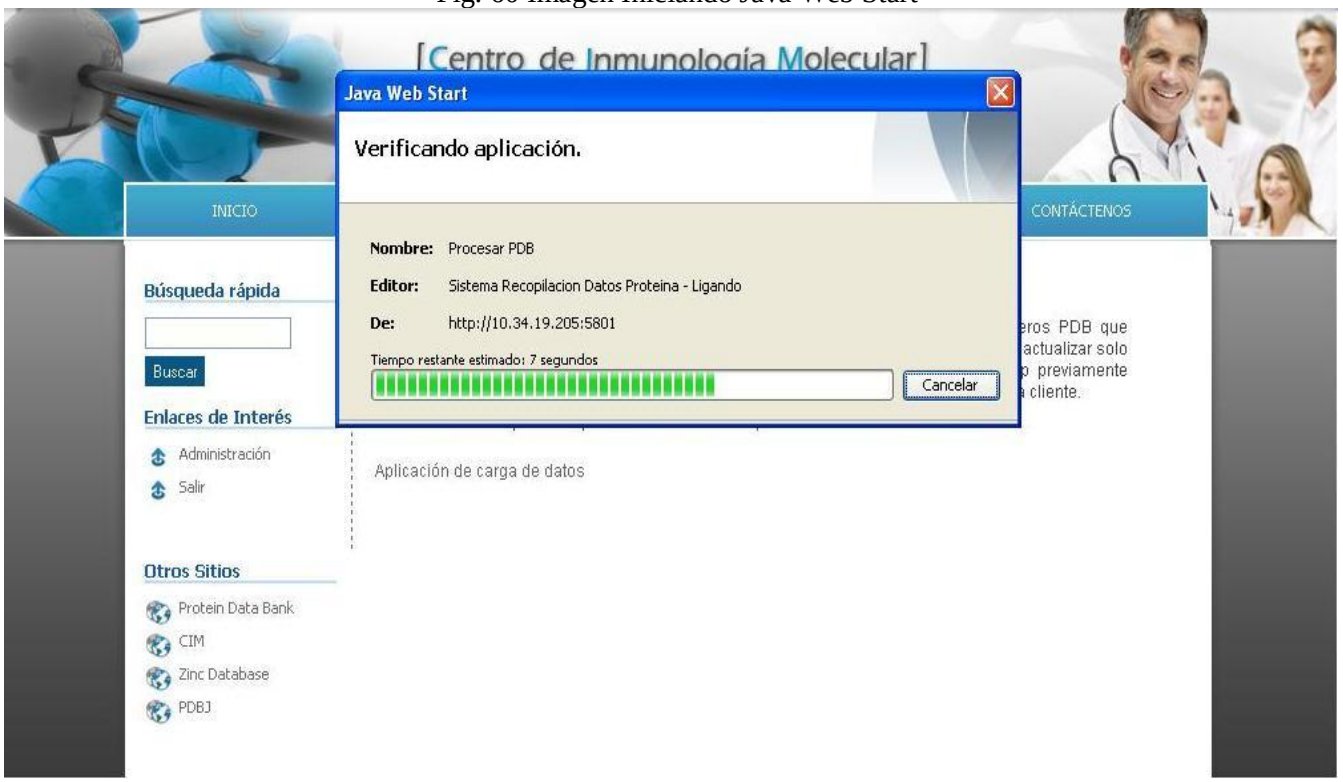


Fig. 61 Imagen Cargando Aplicación Escritorio

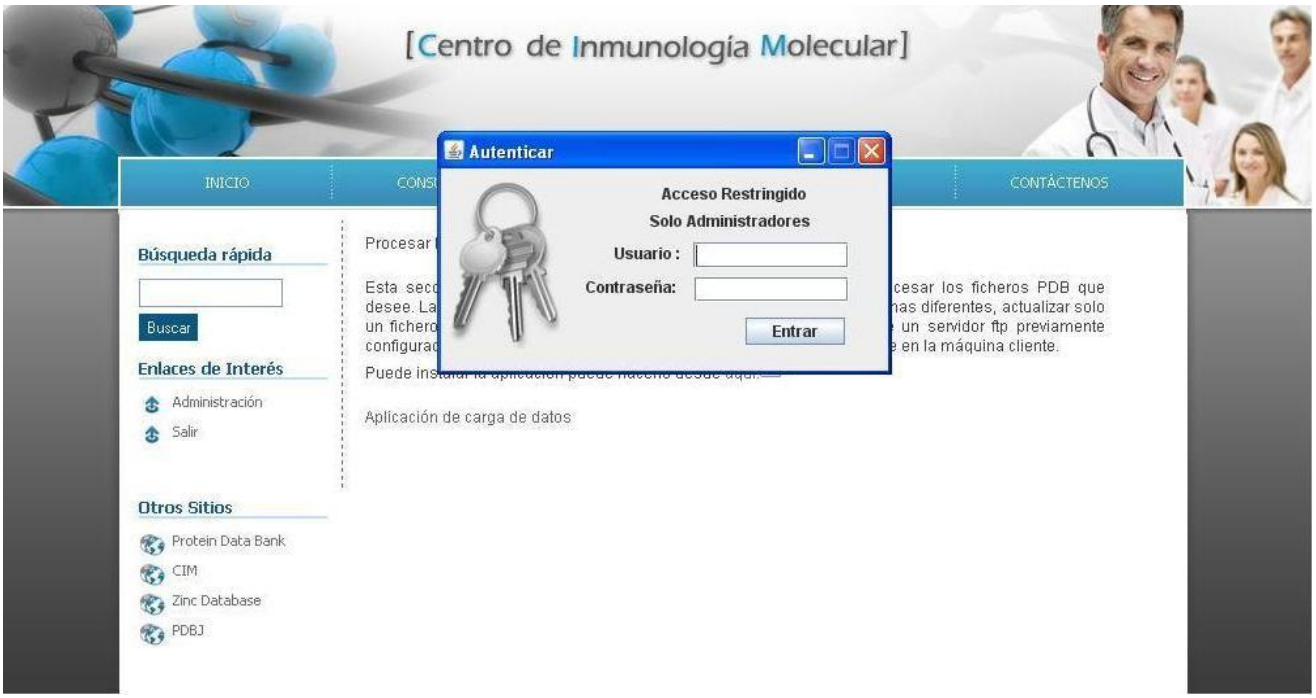


Fig. 62 Imagen Autenticar Administrador Aplicación Escritorio

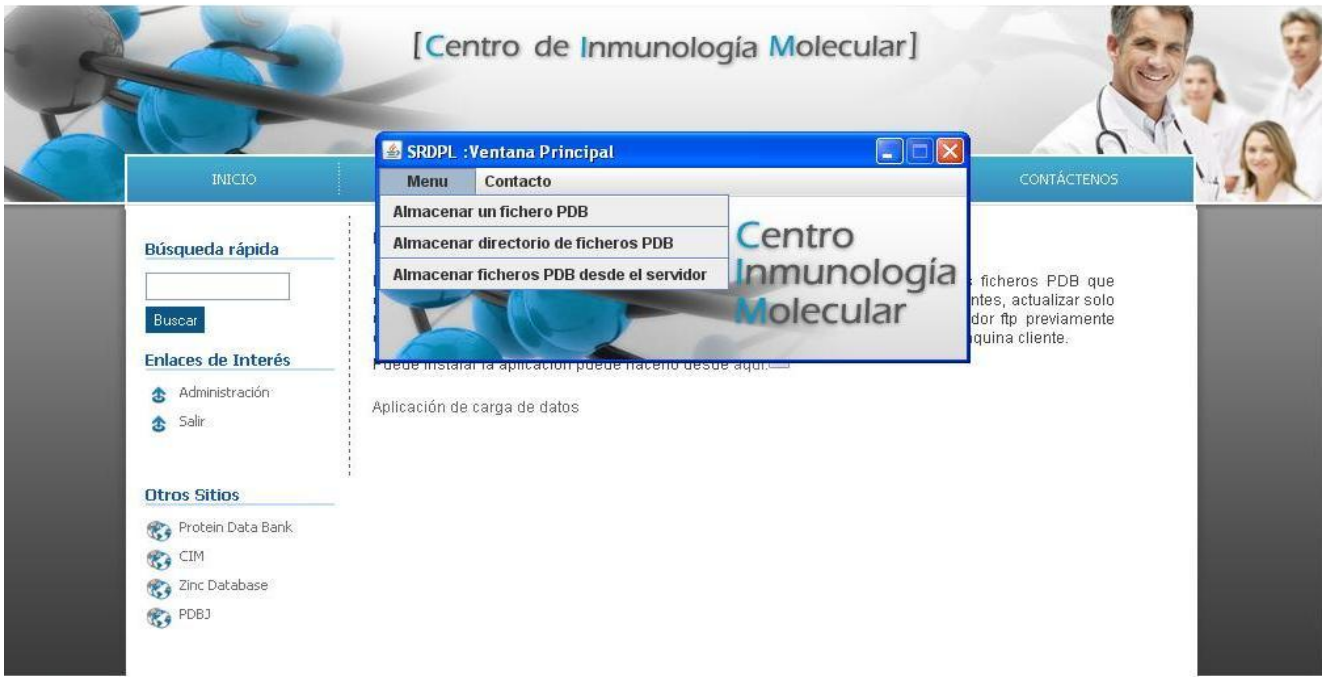


Fig. 63 Imagen Aplicación Escritorio

CONCLUSIONES

En este capítulo se presentó cómo se implementan los elementos del diseño de la aplicación escritorio y Web en término de componentes. Se mostró la vista lógica de la implementación del sistema y los diagramas de componente generados para cada caso de uso. Además se describieron los algoritmos más importantes de la aplicación escritorio y Web. Se mostró cómo fue posible integrar la aplicación escritorio con la aplicación Web y se presentaron un conjunto de imágenes del sistema.

Conclusiones Generales

- ✓ Se realizó un estudio sobre la fuente Protein Data Bank, sobre los mecanismos de funcionamiento de las herramientas ETL (Extract – Transform - Load) y sobre la investigación realizada por el Centro de Inmunología Molecular, con el objetivo de seleccionar las técnicas y herramientas para el desarrollo del trabajo.

- ✓ Partiendo de las funcionalidades identificadas se determinó que para un mejor funcionamiento del sistema quedaría constituido por una aplicación escritorio y una aplicación Web.

- ✓ Se realizó el rediseño de la base de datos inicial obteniéndose una base de datos relacional que cumple con los requerimientos actuales del Centro de Inmunología Molecular y con los del sistema.

- ✓ Se implementó una aplicación escritorio para llevar a cabo el proceso de extracción, transformación y carga de los datos obtenidos de la fuente Protein Data Bank y se implementó una aplicación Web para que pudieran ser consultados.

- ✓ La integración de la aplicación escritorio y la aplicación Web se hizo con la tecnología Java Web Start.

- ✓ Se les realizó el proceso de extracción, transformación y carga a la base de datos a 13 mil archivos PDB de los cuales quedaron almacenados en la base de datos 2774, disponibles para ser consultados.

Recomendaciones

- ✓ Incorporar al sistema la funcionalidad de mantenerlo actualizado directamente desde internet.

- ✓ Implementar la funcionalidad al sistema de hacer el proceso de extracción, transformación y carga de datos de forma distribuida o paralelo, para incrementar la velocidad de procesamiento de los archivos PDB.

- ✓ Estudiar el algoritmo implementado por los investigadores del Centro de Inmunología Molecular para incorporar la lógica de extracción de los datos a las reglas de extracción del sistema y la lógica de transformación de los datos a las reglas de transformación del sistema.

Referencias Bibliográficas

1. MSN Encarta [En línea] [Citado el: 2 de Marzo de 2009] Disponible en: http://es.encarta.msn.com/encyclopedia_761560993/f%C3%A1rmaco.html
2. Ceballos Ortiz, Dayamis; Morell Guerra, Persy. *Implementación de una aplicación web para la gestión de la información en estudios de docking.* Universidad de las Ciencias Informáticas. 2008.
3. Luis A. Diago, Persy Morell, Longendri Aguilera and Ernesto Moreno. Setting up a large set of protein-ligand PDB complexes for the development and validation of knowledge-based docking algorithms [En línea] [Citado el: 1 de Marzo de 2009] Disponible en: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2008766
4. Jmol: un visor Java de código abierto para estructuras químicas en tres dimensiones. [En línea] [Citado el: 2 de Marzo de 2009] Disponible en: http://jmol.sourceforge.net/index.es.html
5. Visor de moléculas en 3D desarrollado en Java [En línea] [Citado el: 3 de Marzo de 2009] Disponible en: http://jmol.uptodown.com/ebay/
6. Metodologías de desarrollo de software [En línea] [Citado el: 3 de Marzo de 2009] http://www.rhernando.net/modules/tutorials/doc/ing/met_soft.html
7. [En línea] [Citado el: 3 de Marzo de 2009] OpenUP como alternativa metodológica para proyectos pequeños de software http://kasyles.blogspot.com/2008/09/openup-como-alternativa-metodolgica.html
8. Paradigma visual para UML (Plataforma Java) (Visual Paradigm for UML [Java Platform]) 6.0 [En línea] [Citado el: 24 de Marzo de 2009] Disponible en: http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5Bcuenta_de_Plataforma_de_Java_14715_p/
9. Introduccion al lenguaje Java [En línea] [Citado el: 6 de Marzo de 2009] Disponible en: http://www.wikilearning.com/curso_gratis/introduccion_al_lenguaje_java-introduccion_a_java/5054-2
10. ¿Qué es un framework? [En línea] [Citado el: 8 de Marzo de 2009] Disponible en: http://americati.com/blog/?p=105
11. Spring Framework [En línea] [Citado el: 8 de Marzo de 2009] Disponible en: http://www.linuxparatodos.net/portal/staticpages/index.php?page=spring-framework
12. ¿Qué es el software de Java Web Start y cómo se ejecuta? [En línea] [Citado el: 9 de Marzo de 2009] Disponible en: http://www.java.com/es/download/faq/java_webstart.xml
13. Hibernate http://www.unife.edu.pe/ing/desarrollo.doc

14. Hibernate [En línea] [Citado el: 12 de Marzo de 2009] http://www.juntadeandalucia.es/xwiki/bin/view/MADEJA/Hibernate
15. Abián, Miguel Ángel. El archipiélago Eclipse. [En línea] [Citado el: 28 de Marzo de 2009] Disponible en http://www.javahispano.org/contenidos/es/el_archipielago_eclipse/
16. Base de Datos PostgreSQL, SQL avanzado y PHP [En línea] [Citado el: 9 de Marzo de 2009] http://www.usabilidadweb.com.ar/postgre.php
17. Comparativa entre Postgres y MySQL? [En línea] [Citado el: 9 de Marzo de 2009] Disponible en: http://www.taringa.net/posts/info/1073108/Comparativa-entre-Postgres-y-MySQL.html
18. Utilización del Patrón Modelo-Vista-Controlador (MVC) en el diseño de software educativos. [En línea] [Citado: el 9 de Marzo de 2009] Disponible en:http://www.monografias.com/trabajos43/patron-modelo-vista/patron-modelo-vista.shtm.
19. Patrones de diseño . [En línea] [Citado: el 9 de Marzo de 2009] Disponible http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php

Bibliografía

BioMed Central .Luis A. Diago, Persy Morell, Longendri Aguilera and Ernesto Moreno. Setting up a large set of protein-ligand PDB complexes for the development and validation of knowledge-based docking algorithms. [En línea] [Citado el: 1 de Marzo de 2009] Disponible en: http://www.biomedcentral.com/1471-2105/8/310/abstract

Molecular Docking Web. [En línea] [Citado el: 1 de Marzo de 2009] Disponible en: http://mgl.scripps.edu/people/gmm/

Protein Data Bank [En línea] [Citado el: 1 de Marzo de 2009] Disponible en: http://www.pdb.org/pdb/home/home.do

PDBj [En línea] [Citado el: 1 de Marzo de 2009] Disponible en: http://www.pdbj.org/

ETL: Extract - Transform - Load (and data management and integration) [En línea] [Citado el: 7. de Marzo de 2009] Disponible en: http://www.kjube.be/tnenopxe/index.php?section=28

Extract, Transform and Load (ETL) [En línea] [Citado el: 7 de Marzo de 2009] Disponible en: http://soaagenda.com/journal/articulos/orquestadores-y-soa/

Proceso ETL [En línea] [Citado el: 7 de Marzo de 2009] Disponible en: http://etl-tools.info/es/bi/proceso_etl.htm

¿ETL o E-LT? [En línea] [Citado el: 7 de Marzo de 2009] Disponible en: http://www.gravitar.biz/index.php/bi/etl_elt/

Extraer, Transformar y Cargar datos facilmente ocupando una herramienta hecha en Java y libre [En línea] [Citado el: 7 de Marzo de 2009] Disponible en: http://www.javaqro.com/javaqro/node/34

Jmol: Open-source molecular visualization and analysis [En línea] [Citado el: 3 de Marzo de 2009] Disponible en: http://chemapps.stolaf.edu/jmol/presentations/confchem2006/jmol-confchem.htm

Unified Modeling Language [En línea] [Citado el: 7 de Marzo de 2009] Disponible en: http://www.org/

Rational Rose Technical Developer [En línea] [Citado el: 7 de Marzo de 2009] Disponible en: http://www-142.ibm.com/software/dre/ecatalog/detail.wss?locale=es_ES&synkey=H515804V91308R94

Rational Software [En línea] [Citado el: 7 de Marzo de 2009] Disponible en: http://www-01.ibm.com/software/rational/

Introduction to OpenUP [En línea] [Citado el: 7 de Marzo de 2009] Disponible en: http://epf.eclipse.org/wikis/openup/

James Rumbaugh IJ, Bococh g. El Lenguaje Unificado de Modelado. Manual de Referencia. Segunda Edición. Addison Wesley; 2007.

JavaServer Pages Technology [En línea] [Citado el: 6 de Marzo de 2009] Disponible en: http://java.sun.com/products/jsp/

Apache Tomcat [En línea] [Citado el: 7 de Marzo de 2009] Disponible en: http://tomcat.apache.org/

Hosting Java TomCat JSP [En línea] [Citado el: 7 de Marzo de 2009] Disponible en: http://www.mejoramos.com/hosting-tomcat-jsp/

¿GlassFish o Tomcat? ¿Cuál le conviene? [En línea] [Citado el: 7 de Marzo de 2009] Disponible en http://www.linux.org.ni/modules.php?name=News&file=article&sid=47

Pressman, Roger S. / Madrid, McGraw-Hill. Ingeniería del Software: un enfoque práctico. Parte I y II [En línea] 2002 [Citado el: diciembre de 2007.] Disponible en: http://bibliodoc.uci.cu/pdf/reg02689.pdf

PARADIGM, V. Documentation [En línea] [Citado: el 9 de Marzo de 2009] Disponible en: http://www.visual-paradigm.com/.

Spring.net. [En línea] [Citado: el 9 de Marzo de 2009] Disponible en: http://www.springframework.net.

Spring: framework de java. [En línea] [Citado: el 9 de Marzo de 2009] Disponible en: http://sentidoweb.com/2006/12/26/spring-framework-de-java.php.

Lago Torres, Manuel. Introducción al diseño con patrones [En línea] [Citado: el 9 de Marzo de 2009] Disponible en: http://www.elrincondelprogramador.com/default.asp?id=29&pag=articulos/leer.asp

Patrones para asignación de responsabilidades. [En línea] [Citado: el 9 de Marzo de 2009] Disponible en: https://s3.amazonaws.com/ppt-download/gonzalorojas-12-uml-patrones-de-diseno1574

Anexos

Centro de Inmunología Molecular

2 de junio de 2009

Aval del trabajo:

"Sistema de recopilación de datos para el análisis de las interacciones proteína – ligando**"**

Autores: Felipe Rodriguez Arias
Yunior Bauta Pentón
Yanet Marrero Vargas

Tutores: MSc. Longendri Aguilera Mendoza, UCI
Dr. Ernesto Moreno Frías, Centro de Inmunología Molecular (CIM)
Ing. Reynaldo Alvarez Luna, UCI

El trabajo realizado por este grupo de estudiantes es la continuación de un proyecto de investigación básica donde colaboran el CIM y la UCI. En esta etapa del proyecto, los estudiantes diseñaron y ejecutaron una nueva implementación, en un servidor PostgreSQL de una base de datos que contiene información estructural sobre miles de complejos proteína-ligando.

La implementación de la base de datos en su nuevo formato siguió un diseño cuidadoso en cuanto a la estructuración de los datos, que permite realizar diferentes tipos de consultas de manera eficiente. Además, los estudiantes desarrollaron procedimientos para actualizar los contenidos de la base de manera automatizada.

El trabajo realizado por los estudiantes tiene un gran **valor práctico**, no solo para el CIM, sino también para otros grupos que trabajan en el campo de la Bioinformática en nuestro país. La base de datos en su nuevo formato constituye una valiosa herramienta, que será utilizada por nuestro grupo en el CIM en el marco de un proyecto de desarrollo de algoritmos de predicción de interacciones proteína-ligando.

Durante la ejecución de este proyecto, los estudiantes mostraron una gran motivación y dedicación al trabajo, que hoy se ve reflejada en una excelente tesis de grado.

Dr. Ernesto Moreno Frías
J' Grupo de Nanobiología, Dirección de Investigaciones
Centro de Inmunología Molecular

Anexo 1. Descripción del Caso de Uso: Autenticar Usuario.

Nombre del CUS	Autenticar Usuario
Actores	Usuario(Inicia el CU)
Propósito	Permitir autenticarse en el sistema.
Resumen	El caso de uso se inicia cuando el usuario selecciona una sección del sistema. El sistema muestra una interfaz para que le usuario introduzca los datos para autenticarse. El sistema verifica que todos los datos necesarios hayan sido insertados y de forma correcta, dando acceso al sistema según el nivel de acceso, finalizando el caso de uso.
Referencias	R1.
Precondiciones	El usuario debe estar registrado en el sistema.
Poscondiciones	Se habilitan los permisos según los privilegios.

Curso Normal de los Eventos	
Acciones del Actor	**Respuesta del Sistema**
1. El usuario selecciona una sección del sistema.	2. El sistema muestra la interfaz para que el usuario inserte el nombre y la contraseña.
3. El usuario inserta los datos.	4. El sistema verifica que los datos hayan sido insertados y de forma correcta.
	5. El sistema brinda acceso al usuario según el nivel de acceso y a la sección solicitada por el usuario.

Flujos Alternos	
Acciones del Actor	**Respuesta del Sistema**
	4.1 Si los datos fueron insertados de forma incorrecta el sistema muestra un mensaje “Ha fallado la autenticación”. Ir al paso 2.
	4.2 Si la autenticación se hace en la sección “Procesar PDB” y es un investigador el que se autentica el sistema muestra el mensaje de error “Usted no tiene acceso a esta sección”.

Prioridad:	Crítico

Anexo 2. Descripción del Caso de Uso: Registrar Investigador.

Nombre del CUS	Registrar Investigador
Actores	Investigador (Inicia el CU)
Propósito	Permitir registrarse en el sistema para poder tener acceso al mismo.
Resumen	El caso de uso inicia cuando el investigador selecciona la opción "Registrarse". El sistema muestra la interfaz correspondiente a la opción. El Investigador realiza las acciones necesarias. Finaliza el caso de uso.
Referencias	R2.
Precondiciones	
Poscondiciones	El investigador queda registrado en el sistema y se le habilitan los permisos.

Sección: "Registrar Investigador"	
Acciones del Actor	**Respuesta del Sistema**
1. El investigador selecciona la opción "Registrarse".	2. El sistema muestra la interfaz para llenar los datos del Investigador a registrar (Nombre de Usuario, Contraseña, Confirmar Contraseña, Nombre Verdadero, Apellidos, Correo Electrónico (opcional).
3. El investigador inserta los datos.	4. El sistema verifica que los datos obligatorios hayan sido insertados y de forma correcta.
	5. El sistema registra el investigador y muestra el mensaje "Gracias por registrarse".
Flujos Alternos	
Acciones del Actor	**Respuesta del Sistema**
	4.1 Si el correo electrónico insertado por el investigador es inválido o el campo de confirmación de la contraseña es incorrecto el sistema muestra un

	mensaje “Campo Inválido” para ambos casos. Ir a paso 2.
	4.2 Si el investigador dejó vacío uno de los campos obligatorios el sistema manda un mensaje “Campo Requerido”. Ir al paso 2.
	4.3 Si el campo “Contraseña” no concuerda con el campo “Repita contraseña” el sistema muestra el mensaje “Las contraseñas no concuerdan”. Ir al paso 2.
	4.4 Si el nombre del investigador ya existe en la base de datos, el sistema muestra el mensaje “EL nombre de usuario ya existe”. Ir al paso 2.
	4.5 Si el usuario del investigador no excede de cinco caracteres el sistema muestra el mensaje "Inválido, menos de 5 caracteres" o si excede de cuarenta caracteres el sistema muestra el mensaje "Inválido, menos de 40 caracteres".
	4.6 Si la contraseña del investigador no excede de cinco caracteres el sistema muestra el mensaje "Inválido, menos de 5 caracteres" o si excede de veinte caracteres el sistema muestra el mensaje "Inválido, menos de 20 caracteres".
	4.7 Si el nombre real del investigador no excede de dos caracteres el sistema muestra el mensaje "Inválido, menos de 2 caracteres" o si excede de cuarenta caracteres el sistema muestra el mensaje "Inválido, menos de 40 caracteres".
	4.8 Si el apellido del investigador no excede de cuatro caracteres el sistema muestra el mensaje "Inválido, menos de 4 caracteres" o si excede de cuarenta caracteres el sistema muestra el mensaje "Inválido, menos de 40 caracteres".

	4.9 Si el correo electrónico del investigador no excede de cuatro caracteres el sistema muestra el mensaje "Inválido, menos de 4 caracteres" o si excede de cuarenta caracteres el sistema muestra el mensaje "Inválido, menos de 40 caracteres".
Prioridad:	Crítico

Anexo 3. Descripción del Caso de Uso: Visualizar Proteína.

Nombre del CUS	Visualizar proteína
Actores	Usuario (Inicia el CU).
Propósito	Visualizar proteína.
Resumen	El caso de uso se inicia cuando Usuario selecciona la opción “JMOL”. El sistema muestra la interfaz que permitirá al usuario interactuar con el programa JMOL. Finaliza el caso de uso.
Referencias	R3.
Precondiciones	El usuario debe estar registrado en el sistema.
Prioridad:	Crítico

Anexo 4. Descripción del Caso de Uso: Gestionar Usuarios.

Nombre del CUS	Gestionar Usuarios
Actores	Administrador (Inicia el CU)
Propósito	Permitir ver el listado de los usuarios registrados en el sistema, eliminar usuarios y crear un investigador como nuevo administrador.
Resumen	El caso de uso inicia cuando el administrador selecciona la opción “Listar Usuarios” en Administración. El sistema muestra la interfaz correspondiente a la opción. El administrador realiza las acciones necesarias. Finaliza el caso de uso.
Referencias	R 7.1, R 7.2, R 7.3.
Precondiciones	El administrador debe estar registrado en el sistema.
Sección “General”	
Acciones del Actor	**Respuesta del Sistema**

1. El administrador del sistema se dirige al menú principal y selecciona la opción “Administración”.	2. El sistema muestra la interfaz del área de administración con las posibles acciones a realizar por el administrador.
Sección: “Listar Usuarios”	
Acciones del Actor	**Respuesta del Sistema**
3. El administrador selecciona la opción “Listar Usuarios”.	4. El sistema muestra un listado con el usuario, nombre y correo electrónico correspondiente a los usuarios registrados en el sistema. Además brinda la posibilidad de eliminar un usuario y de crear un investigador como administrador.
Sección: “Eliminar Usuario”	
Acciones del Actor	**Respuesta del Sistema**
5. El administrador indica eliminar usuario.	6. El sistema verifica el usuario a eliminar.
	7. El sistema elimina el usuario.
	8. El sistema muestra la lista de los investigadores actualizada.
Sección: “Crear Nuevo Administrador”	
Acciones del Actor	**Respuesta del Sistema**
5. El administrador selecciona “Nuevo Administrador”.	6. El sistema inserta el investigador seleccionado por el administrador como nuevo administrador del sistema.
Flujo Alterno	
Sección: “Eliminar Usuario”	
Acciones del Actor	**Respuesta del Sistema**
	6.1. Si el usuario a eliminar es el único administrador del sistema, el sistema muestra un mensaje “El usuario no se puede eliminar, único administrador”.
	6.2. Si el identificador el usuario a eliminar no se encuentra en la base de datos, el sistema muestra el mensaje “El identificador no existe o no es un

	entero".
Prioridad:	Crítico

Anexo 5. Descripción del Caso de Uso: Gestionar Comentarios del Administrador.

Nombre del CUS	Caso de Uso: Gestionar Comentarios del Administrador.
Actores	Administrador (Inicia el CU)
Propósito	Permitir al administrador ver y responder los comentarios, las inquietudes y sugerencias de los investigadores.
Resumen	El caso de uso se inicia cuando el administrador selecciona la opción "Listar comentarios" en Administración. El sistema muestra las interfaces correspondientes al caso de uso. El administrador realiza las acciones necesarias. Finaliza el caso de uso.
Referencias	R 8.1, R 8.2, R 8.3.
Precondiciones	El administrador debe estar registrado en el sistema.
Curso Normal de los Eventos	
Sección "General"	
Acciones del Actor	**Respuesta del Sistema**
1. El administrador del sistema se dirige al menú principal y selecciona la sección "Administración".	2. El sistema muestra la interfaz del área de administración con las posibles acciones a realizar por el administrador.
Sección "Listar Comentarios"	
Acciones del Actor	**Respuesta del Sistema**
3. El administrador selecciona la opción "Listar Comentarios".	4. El sistema muestra el listado de los comentarios por el título, parte del comentario, muestra si ya fue leído o no. El sistema brinda, además, la posibilidad de ver el comentario completo y de eliminar el comentario.
Sección "Eliminar Comentario"	
Acciones del Actor	**Respuesta del Sistema**
5. El administrador indica eliminar comentario.	6. El sistema verifica el comentario a eliminar.

	7. El sistema elimina el comentario.
	8. El sistema muestra el listado de comentarios actualizado.
Sección "Responder Comentario"	
Acciones del Actor	**Respuesta del Sistema**
5. El administrador selecciona "Ver el mensaje".	6. El sistema muestra la interfaz que muestra el comentario completo y el título del investigador que lo envió y los campos para que el administrador inserte el título y el comentario.
7. El administrador indica respuesta de comentario.	8. El sistema verifica el comentario.
	9. El sistema envía el comentario.
Flujos Alternos	
Sección "Eliminar Comentario"	
Acciones del Actor	**Respuesta del Sistema**
	6.1. Si el sistema no encuentra el comentario a eliminar en la base de datos muestra el mensaje "El identificador no existe o no es un entero".
Sección "Responder Comentario"	
Acciones del Actor	**Respuesta del Sistema**
	8.1 Si el campo del título del comentario está vacío el sistema muestra el mensaje "Debe especificar un Título". Ir al paso 6.
	8.2 Si el campo del comentario está vacío el sistema muestra el mensaje "Debe especificar un Comentario". Ir al paso 6.
	8.3 Si el título del comentario excede los 15 caracteres el sistema muestra el mensaje "El título no debe exceder los 15 caracteres". Ir al paso 6.
Prioridad	Crítico

Anexo 6. Descripción del Caso de Uso: Gestionar Comentarios del Investigador.

Nombre del CUS	Gestionar Comentarios del Investigador.
Actores	Investigador (Inicia el CU)
Propósito	Permitir a los investigadores dejar registrados sus comentarios, las inquietudes y sugerencias sobre la aplicación, así como tener la posibilidad de ver la respuesta del administrador a los mismos.
Resumen	El caso de uso se inicia cuando investigador selecciona la opción "Contáctenos" o "Ver mis mensajes" desde cualquier lugar de la aplicación. El sistema muestra las interfaces correspondientes al caso de uso. El investigador realiza las acciones necesarias, finalizando el caso de uso.
Referencias	R 9.1, R 9.2, R 9.3, R 9.4.
Precondiciones	El investigador debe estar registrado en el sistema.
Poscondiciones	Quedan registrados los comentarios del investigador en el sistema.
Curso Normal de los Eventos	
Sección "Redactar nuevo comentario"	
Acciones del Actor	**Respuesta del Sistema**
1. El investigador selecciona la opción "Nuevo Comentario" desde cualquier lugar de la aplicación.	2. El sistema muestra la interfaz para que el investigador inserte su comentario.
3. El investigador inserta su comentario.	4. El sistema verifica que no se encuentre ningún campo vacío.
	5. El sistema muestra el mensaje "Comentario Enviado".
Sección "Listar Comentarios Respondidos"	
Acciones del Actor	**Respuesta del Sistema**
1. El investigador selecciona la opción "Mis Mensajes" desde cualquier lugar de la aplicación.	2. El sistema muestra la interfaz que permite ver los mensajes respondidos. Además brinda la posibilidad de eliminar el comentario y de responderlo.
Sección "Eliminar Comentario Respondido"	
Acciones del Actor	**Respuesta del Sistema**

3. El Investigador indica eliminar un comentario respondido.	4. El sistema elimina el comentario.
	5. El sistema muestra la lista de los comentarios respondidos actualizada.
Sección "Responder Comentario al Administrador"	
Acciones del Actor	**Respuesta del Sistema**
3. El investigador indica un comentario.	4. El sistema muestra la interfaz que muestra el comentario y brinda la posibilidad de responder el comentario.
5. El investigador indica responder el comentario. Ir al paso 4.	
Flujos Alternos	
Sección "Redactar nuevo comentario"	
Acciones del Actor	**Respuesta del Sistema**
	4.1 Si el campo del título del comentario está vacío el sistema muestra el mensaje "Debe especificar un Título". Ir al paso 2.
	4.2 Si el campo del comentario está vacío el sistema muestra el mensaje "Debe especificar un Comentario". Ir al paso 2.
	4.3 Si el título del comentario excede los 15 caracteres el sistema muestra el mensaje "El título no debe exceder los 15 caracteres". Ir al paso 6.
Prioridad	Crítico

Anexo 6. Descripción del Caso de Uso: Autenticar Administrador.

Nombre del CUS	Autenticar Administrador
Actores	Usuario(Inicia el CU)
Propósito	Permitir al administrador del sistema acceder a la aplicación escritorio.
Resumen	El caso de uso se inicia cuando el administrador indica acceder a la

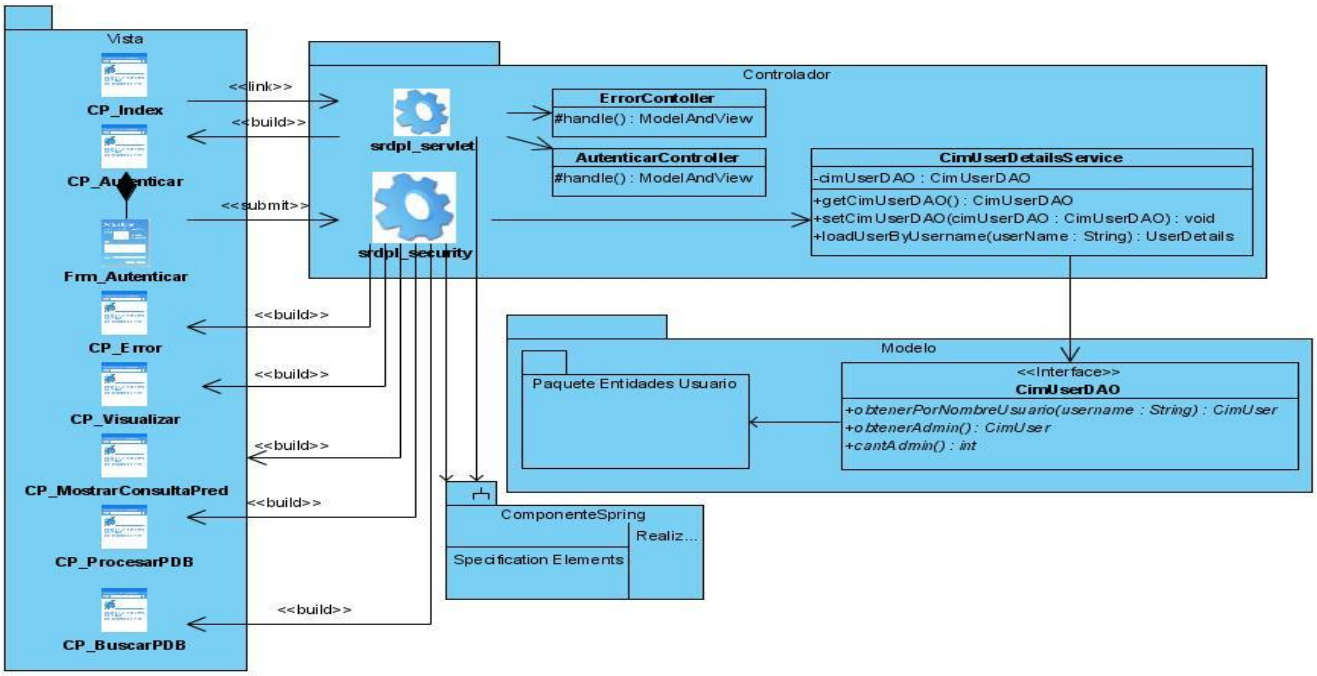
	aplicación de escritorio. El sistema muestra una ventana para que el administrador introduzca los datos para autenticarse. El sistema verifica que todos los datos necesarios hayan sido insertados y de forma correcta, dando acceso al sistema y finalizando el caso de uso.
Referencias	R10.
Precondiciones	El administrador debe estar registrado en el sistema.
Poscondiciones	Se habilitan los permisos para almacenar ficheros PDB en la Base de Datos.

Curso Normal de los Eventos	
Acciones del Actor	**Respuesta del Sistema**
1. El administrador indica autenticarse.	2. El sistema muestra una ventana para que el administrador inserte el usuario y la contraseña.
3. El administrador inserta los datos.	4. El sistema verifica los campos de autenticación.
	5. El sistema verifica que los datos sean correctos.
	6. El sistema brinda acceso al administrador.

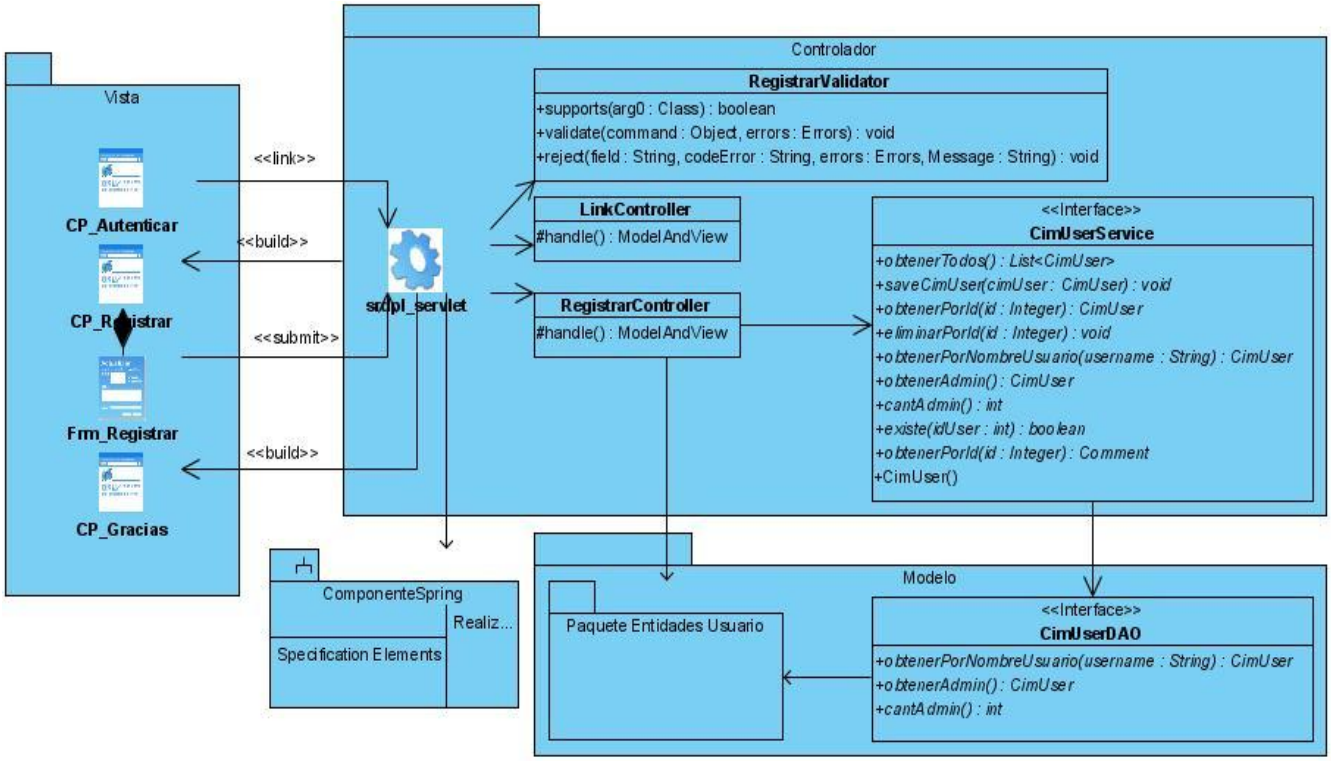
Flujos Alternos	
Acciones del Actor	**Respuesta del Sistema**
	4.1. Si se quedo algún campo sin llenar el sistema muestra el mensaje “Los campos no deben estar vacios”.
	5.1. Si el sistema no encuentra el nombre del administrador registrado en la base de datos muestra el mensaje “El administrador no existe”.
	5.1 Si la contraseña es incorrecta el sistema muestra el mensaje “Contraseña Incorrecta”.
	5.1. Si no es administrador el sistema muestra el mensaje “El usuario no tiene privilegios administrativos”.

Prioridad:	Crítico

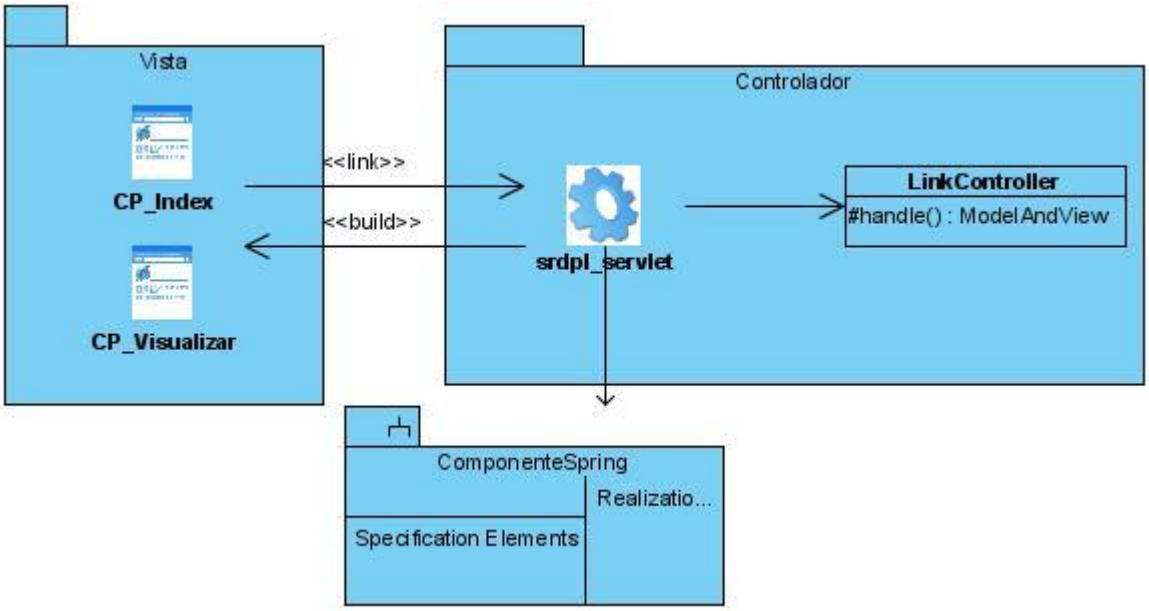
Anexo 7. DCDCU Autenticar Usuario.



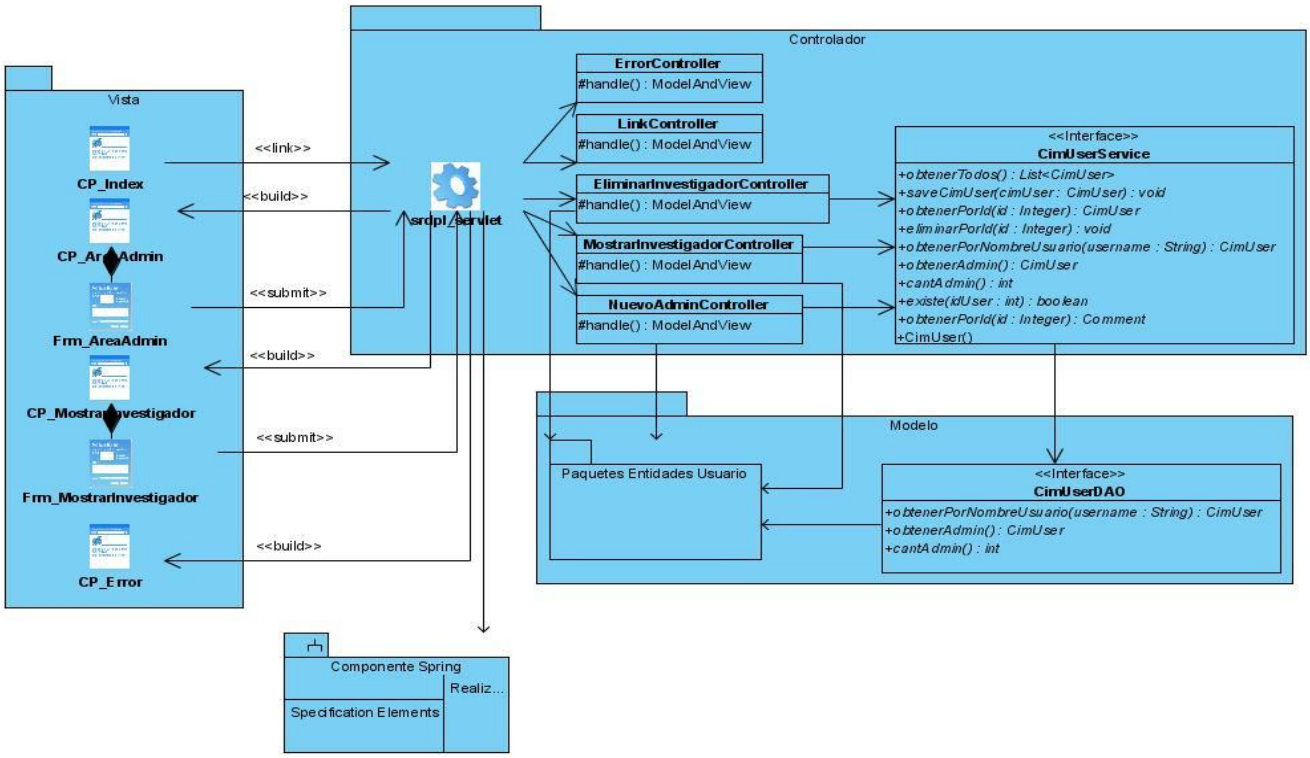
Anexo 8. DCDCU Registrar Investigador.



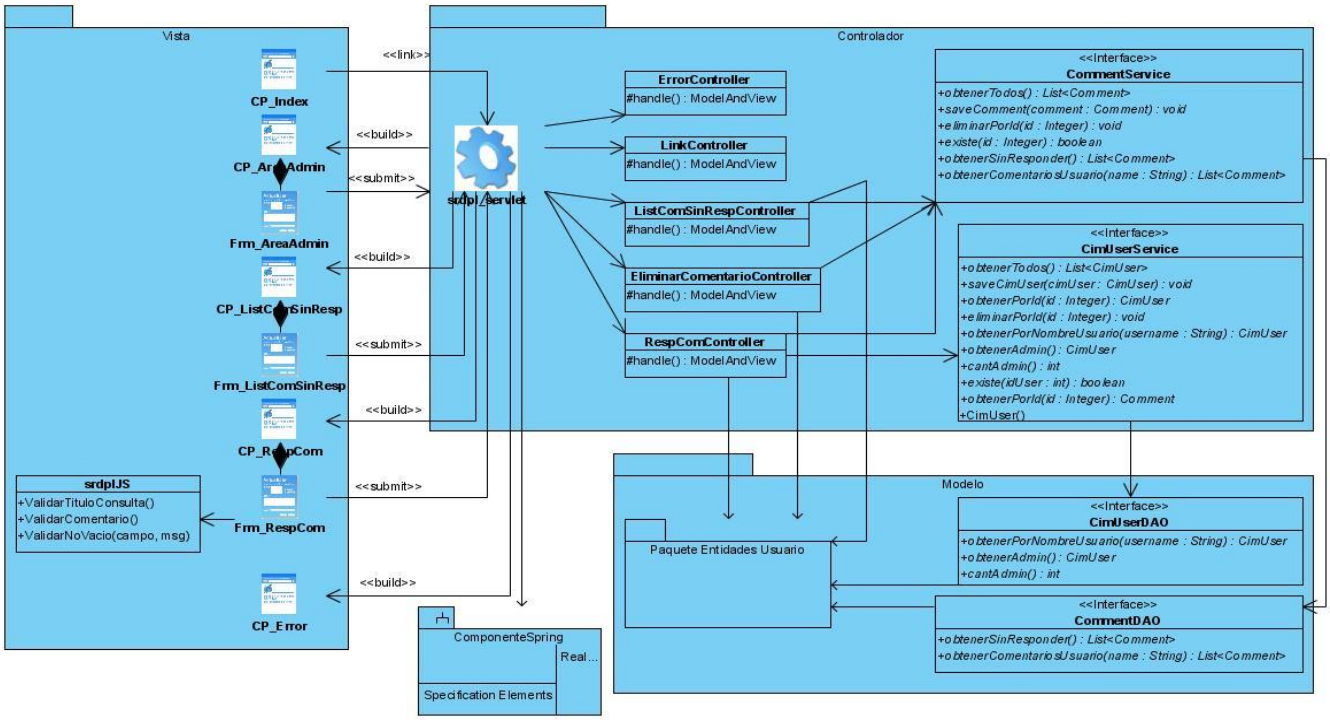
Anexo 9. DCDCU Visualizar Proteína.



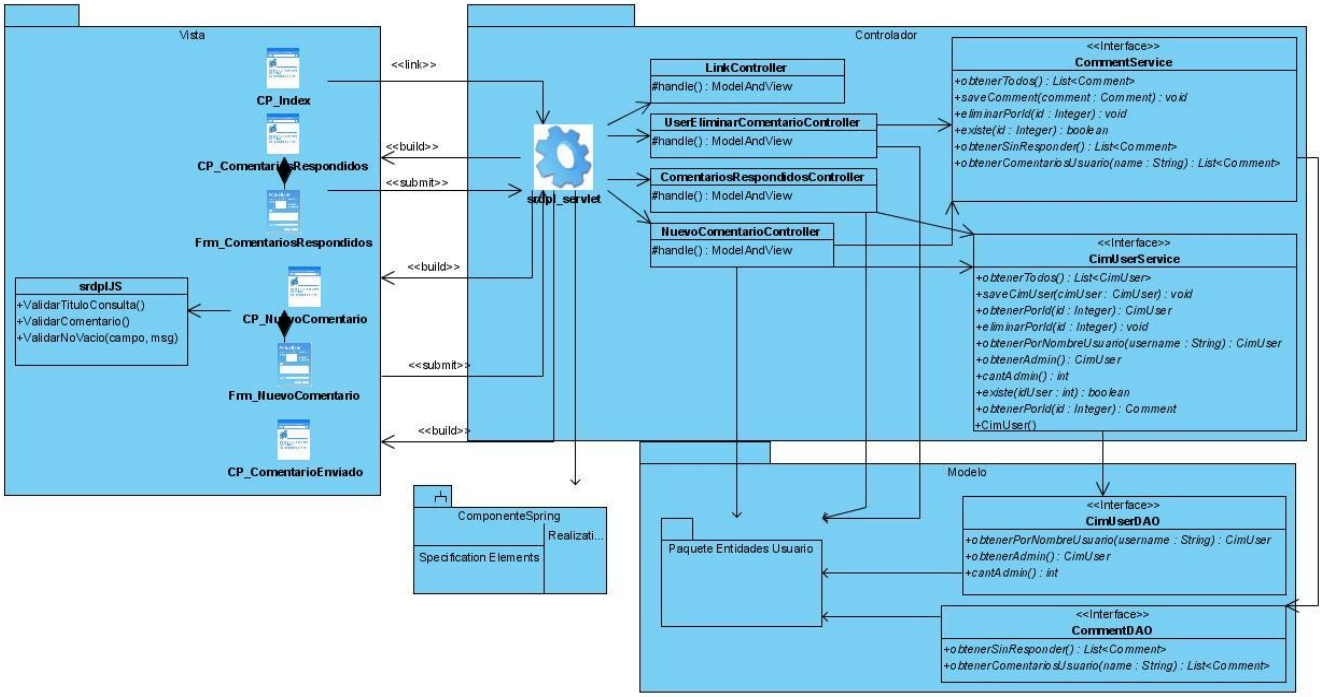
Anexo 10. DCDCU Gestionar Usuarios



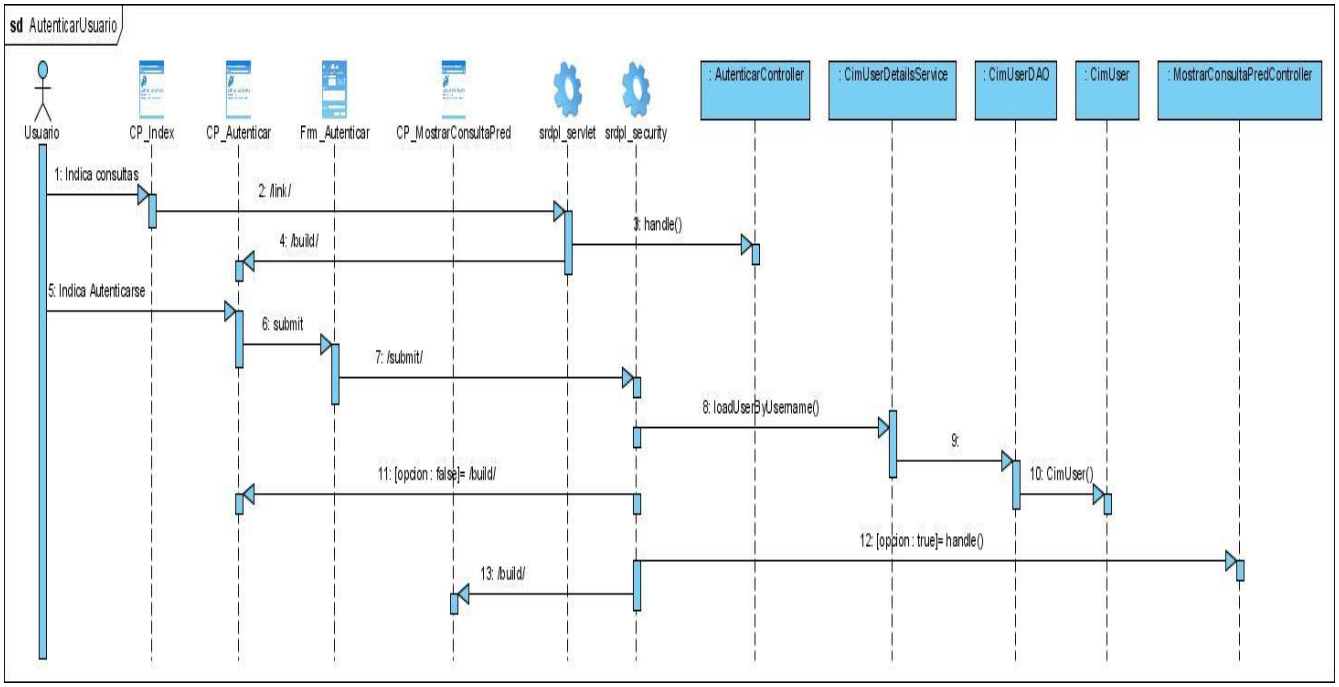
Anexo 11. DCDCU Gestionar Comentarios del Administrador.



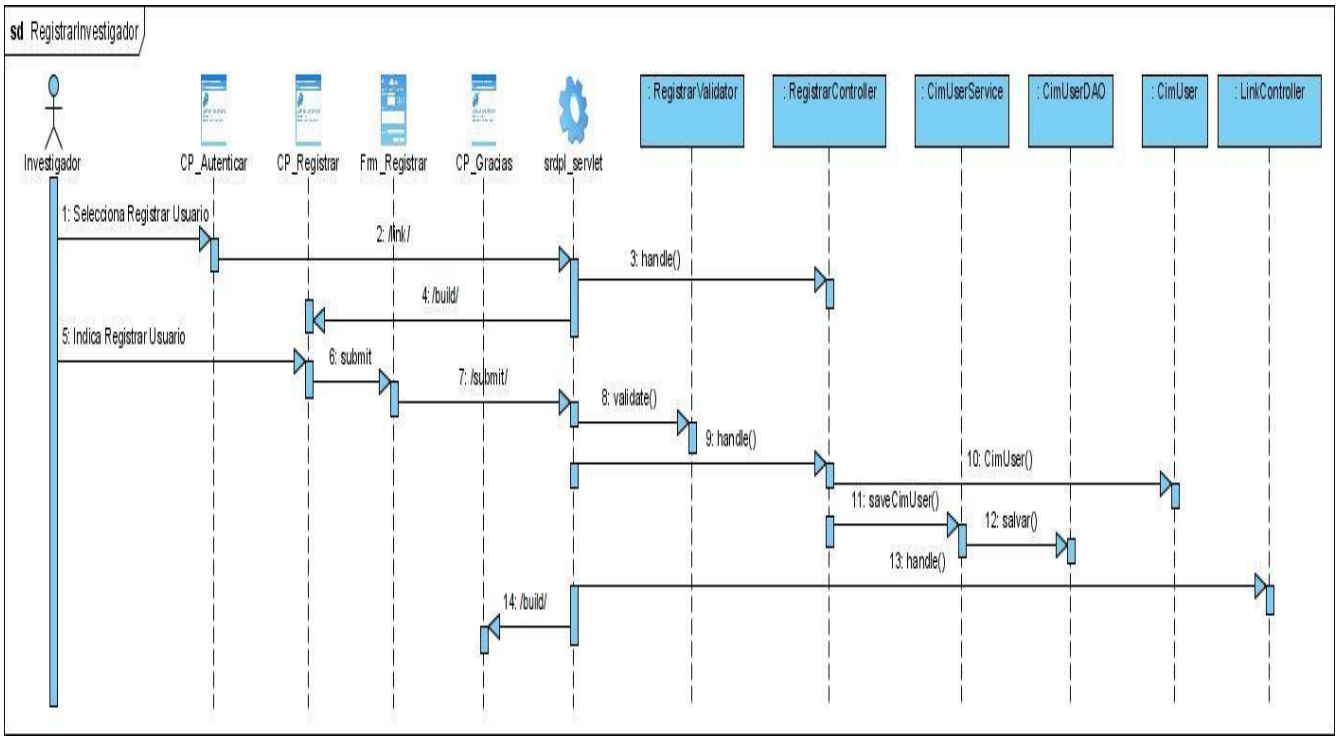
Anexo 12. DCDCU Gestionar Comentarios del Investigador.



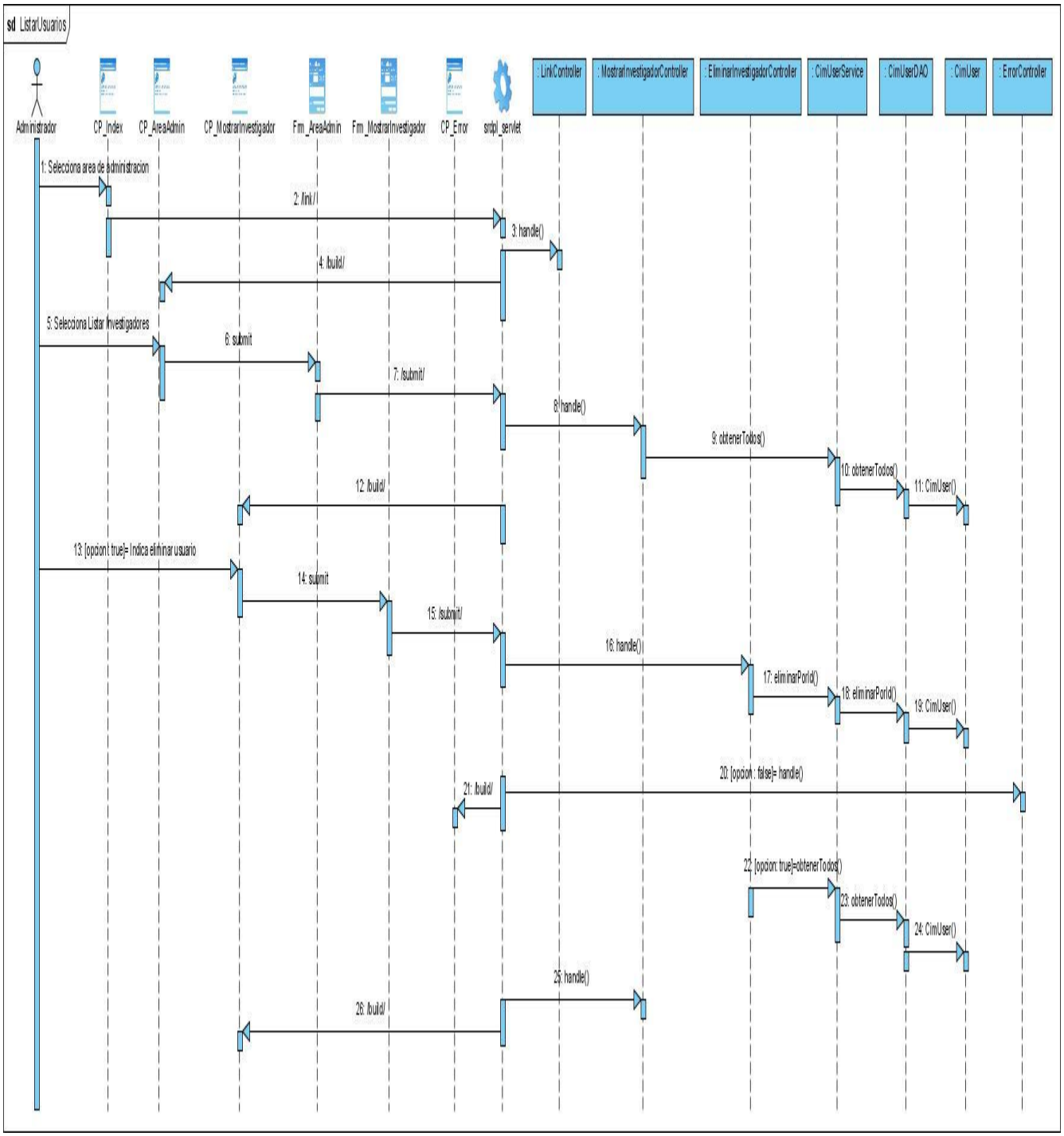
Anexo 13. Diagrama de secuencia CU Autenticar Usuario.



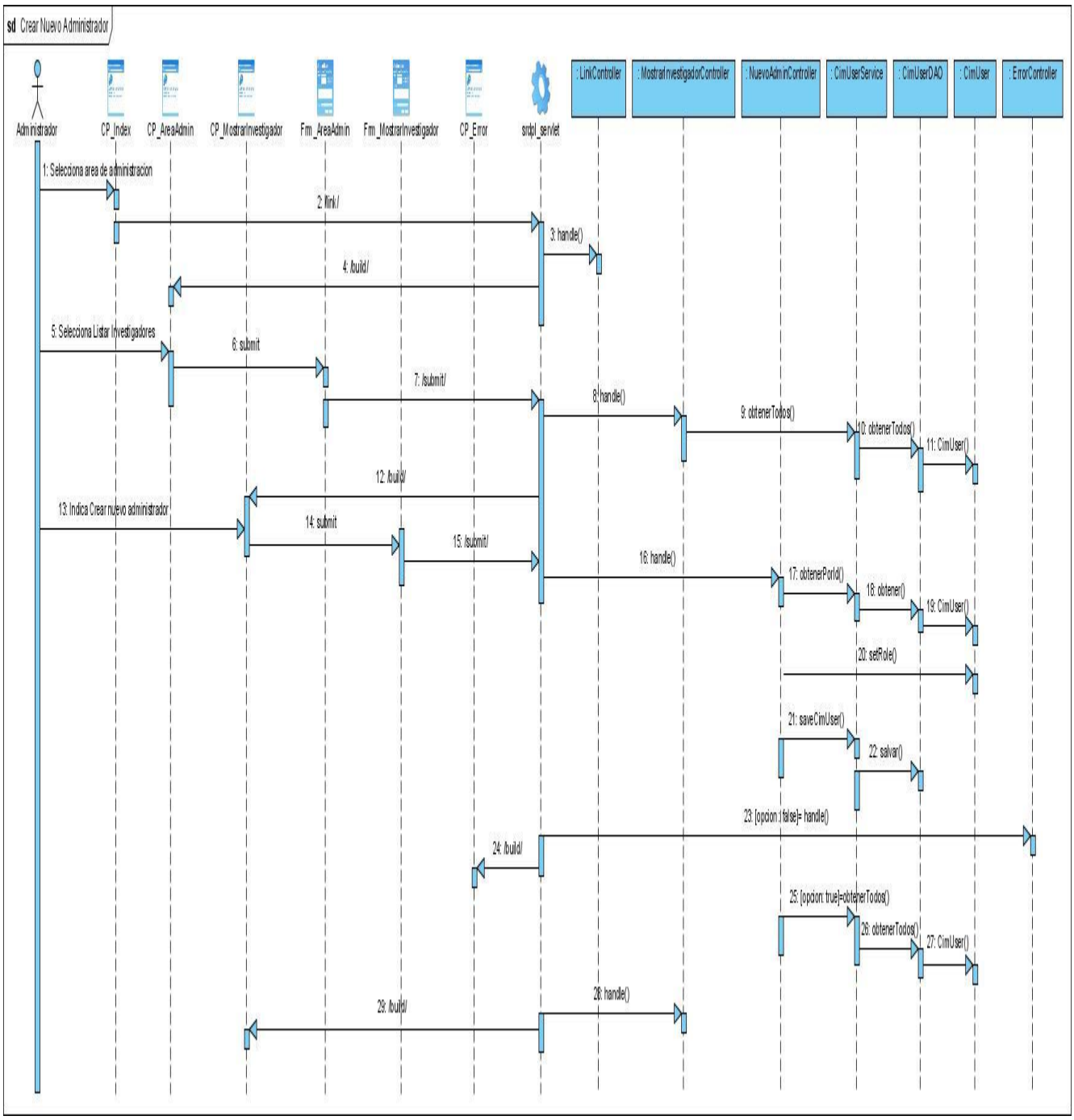
Anexo 14. Diagrama de secuencia CU Registrar Investigador.



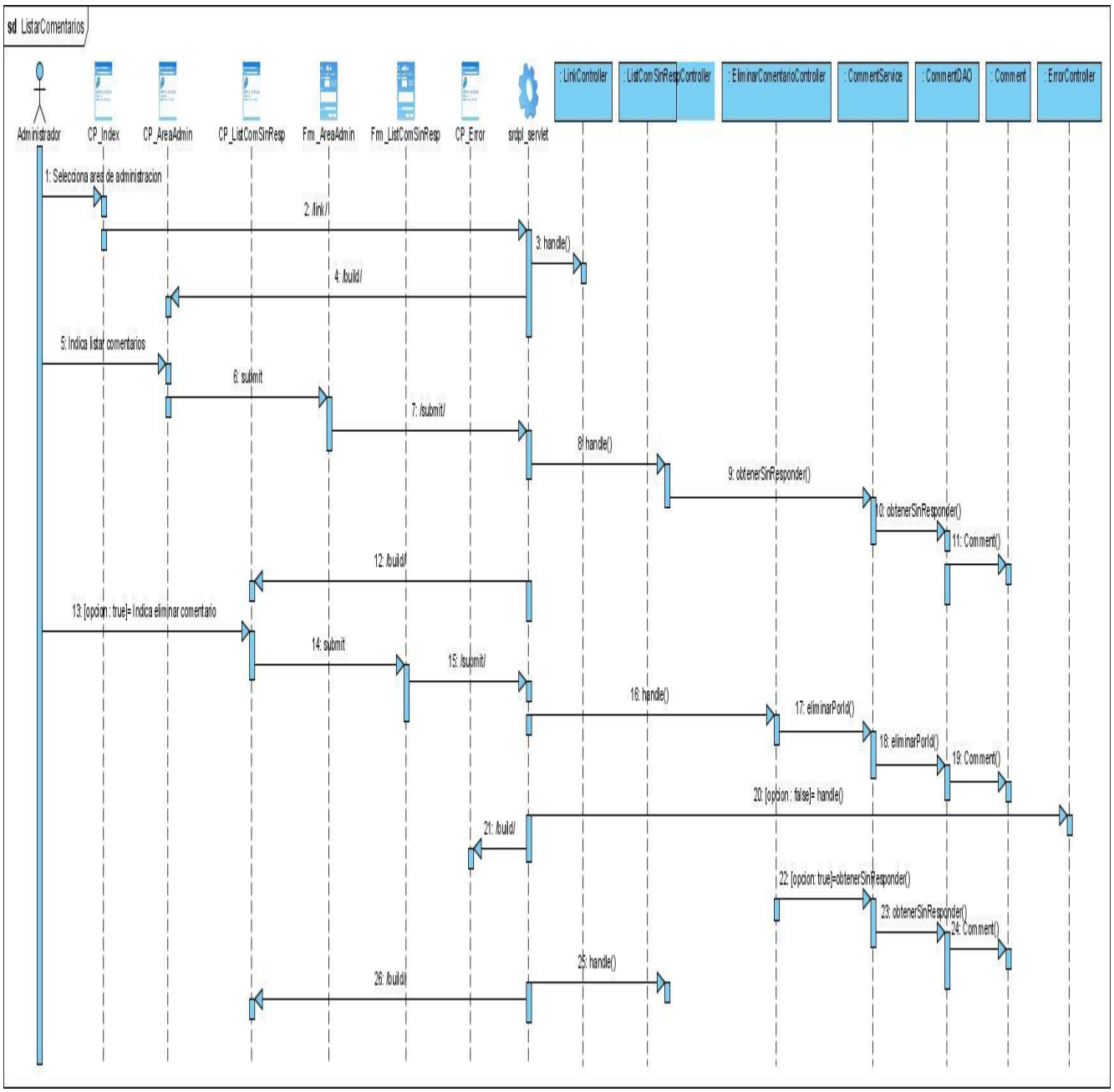
Anexo 15. Diagrama de secuencia CU Gestionar Usuarios (Escenario Listar Usuarios).



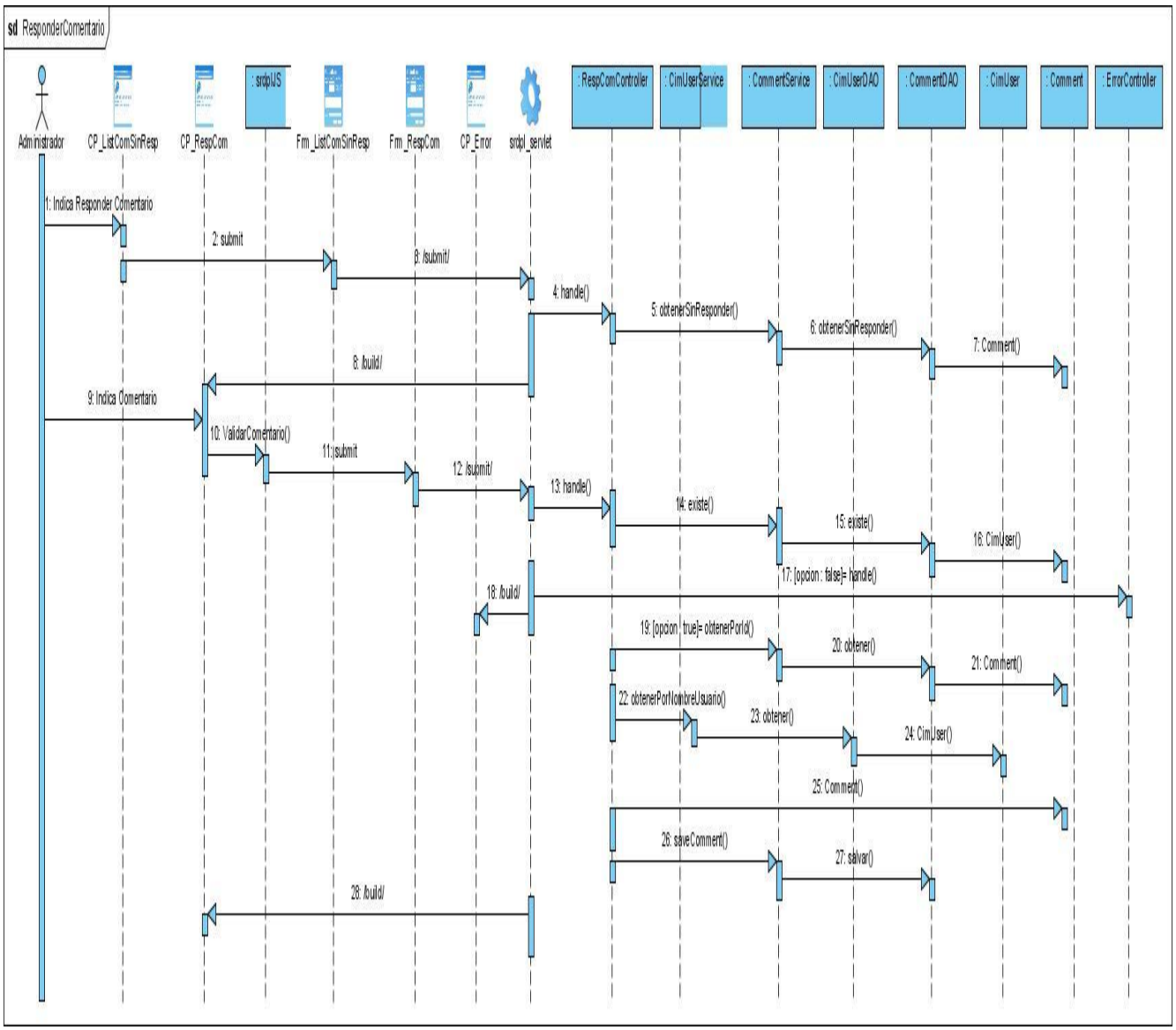
Anexo 16. Diagrama de secuencia CU Gestionar Usuarios (Escenario Crear Nuevo Administrador).



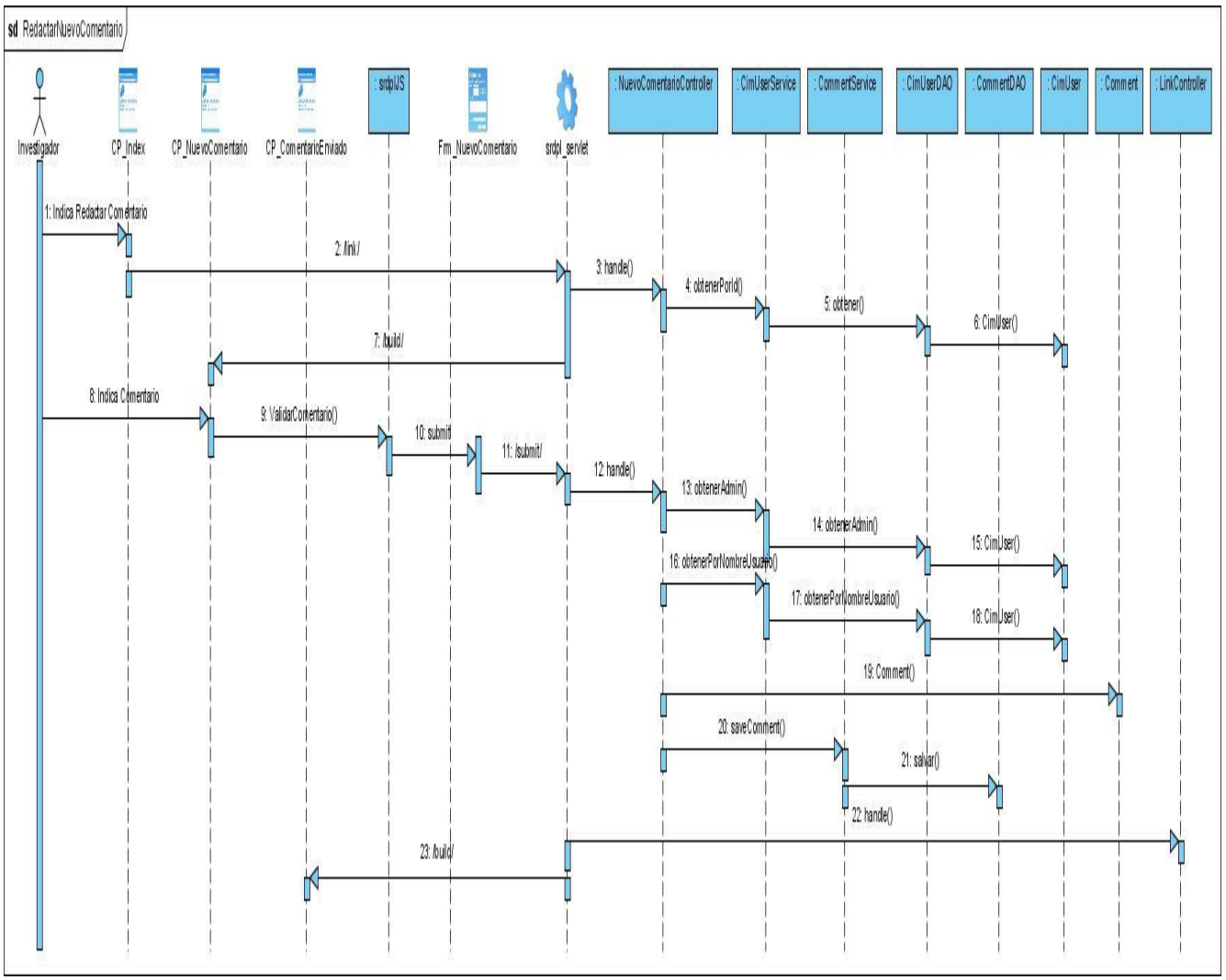
Anexo 17. Diagrama de secuencia CU Gestionar Comentarios del Administrador (Escenario Listar Comentarios).



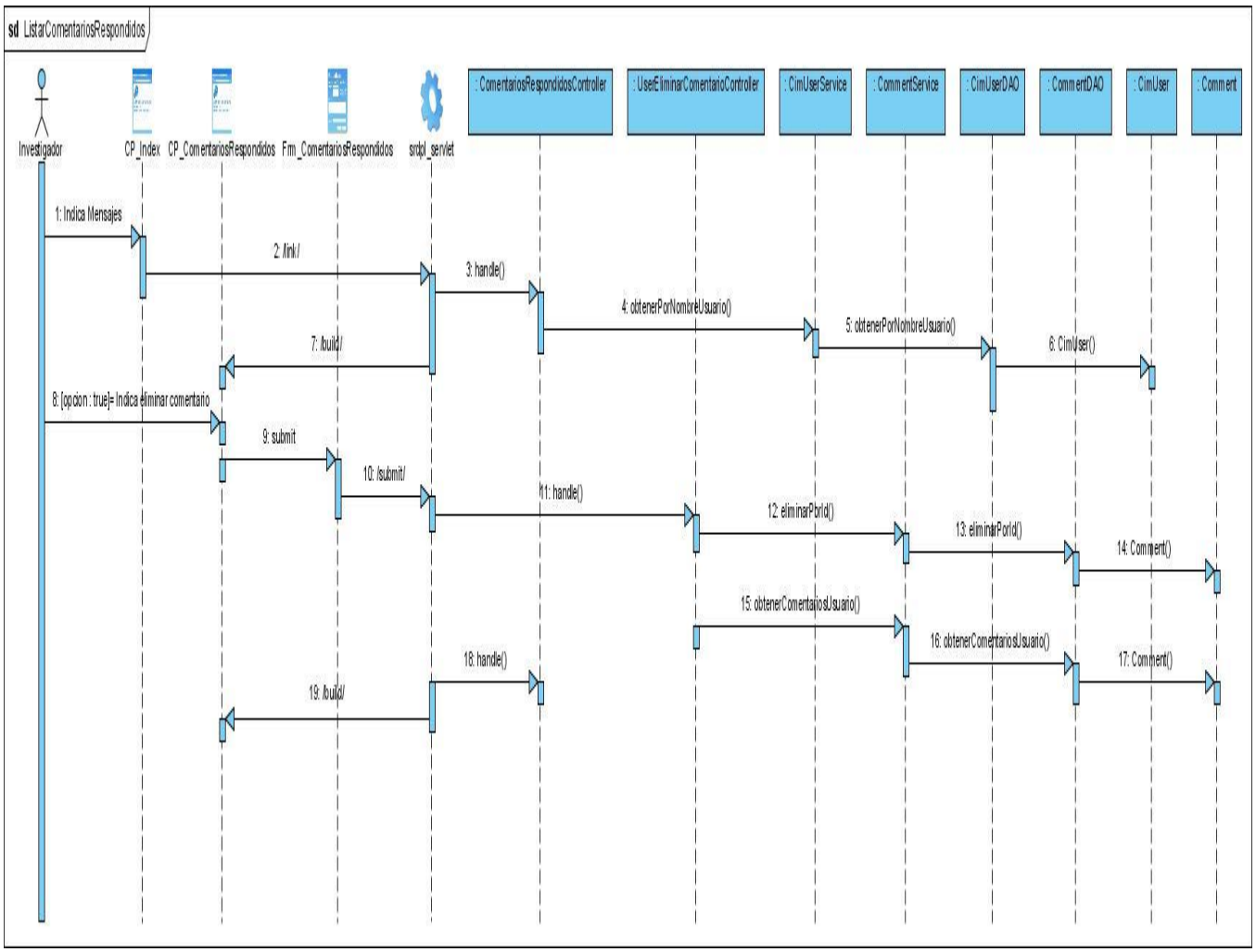
Anexo 18. Diagrama de secuencia CU Gestionar Comentarios del Administrador (Escenario Responder Comentario).



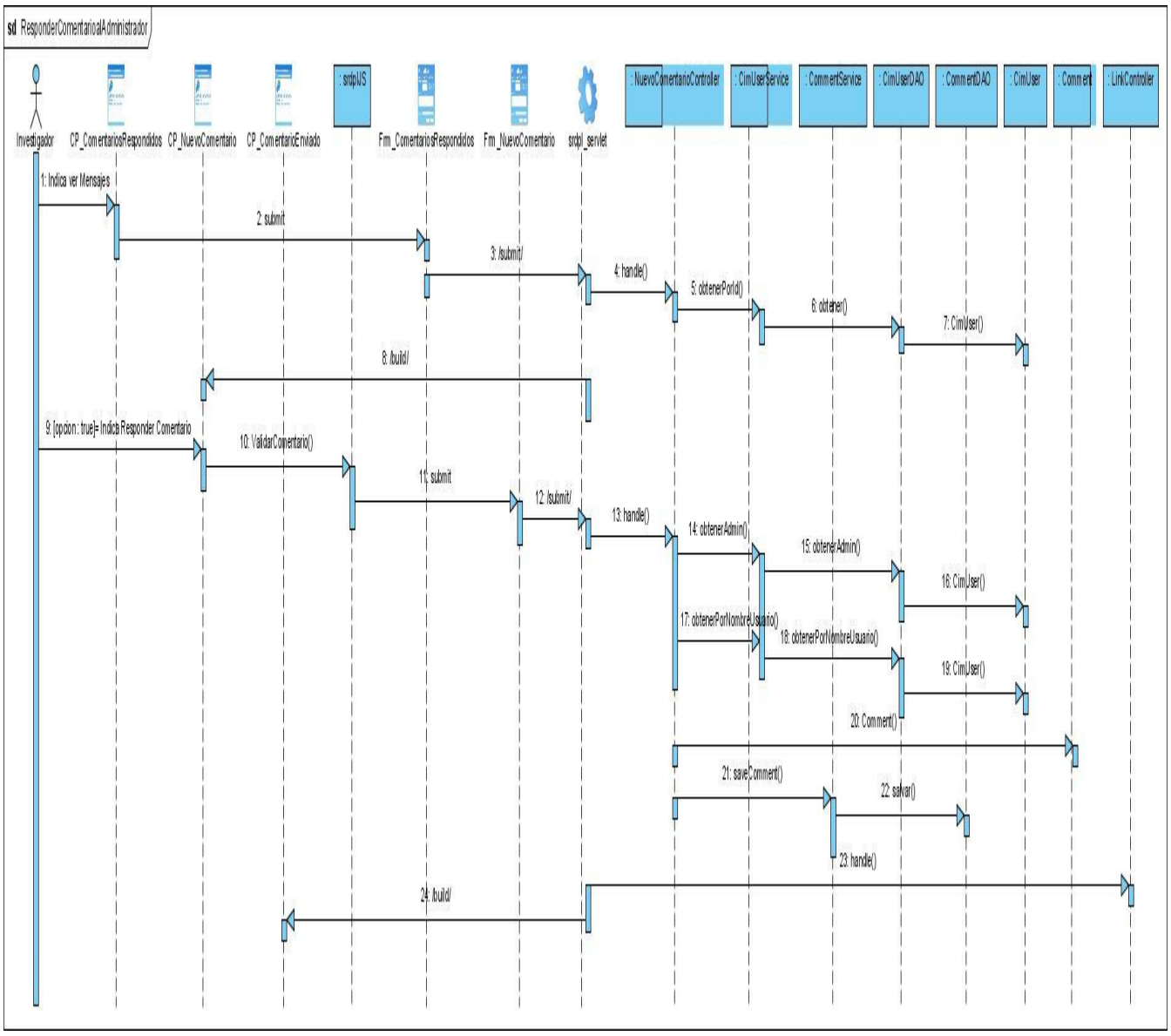
Anexo 19. Diagrama de secuencia CU Gestionar Comentarios del Investigador (Escenario Redactar Nuevo Comentario).



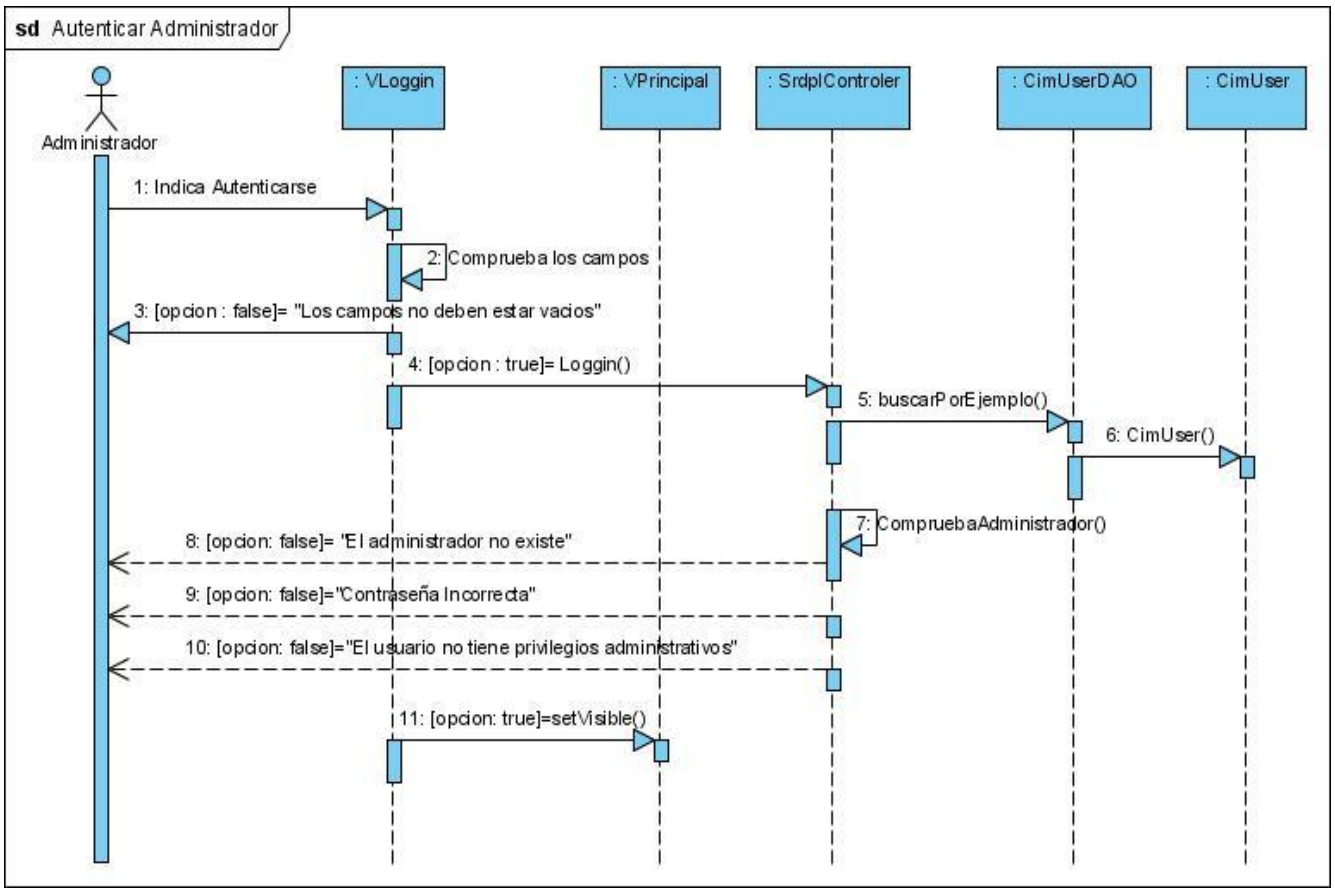
Anexo 20. Diagrama de secuencia CU Gestionar Comentarios del Investigador (Escenario Listar Comentarios Respondidos).



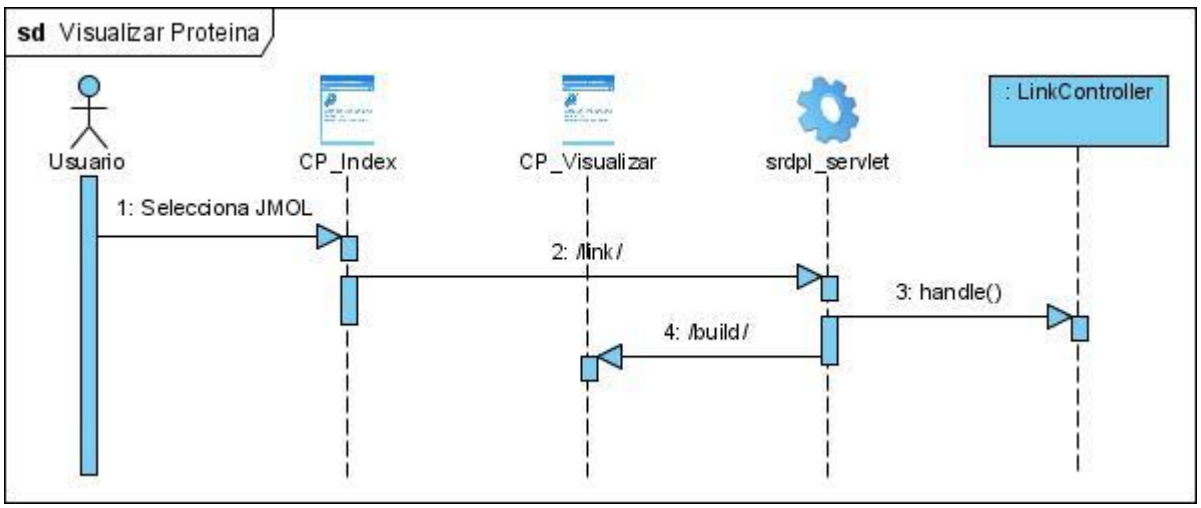
Anexo 21. Diagrama de secuencia CU Gestionar Comentarios del Investigador (Escenario Responder Comentario al Administrador).



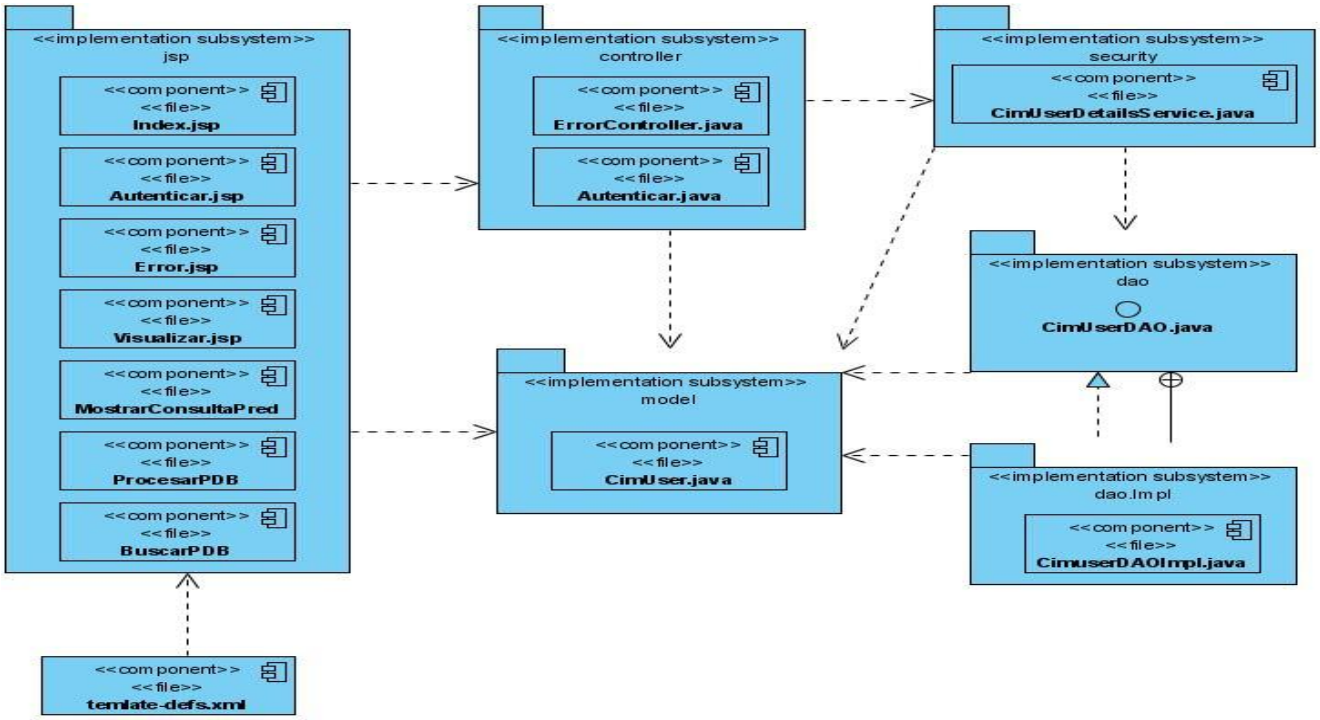
Anexo 22. Diagrama de secuencia CU Actualizar Base de Datos (Escenario Autenticar Administrador)



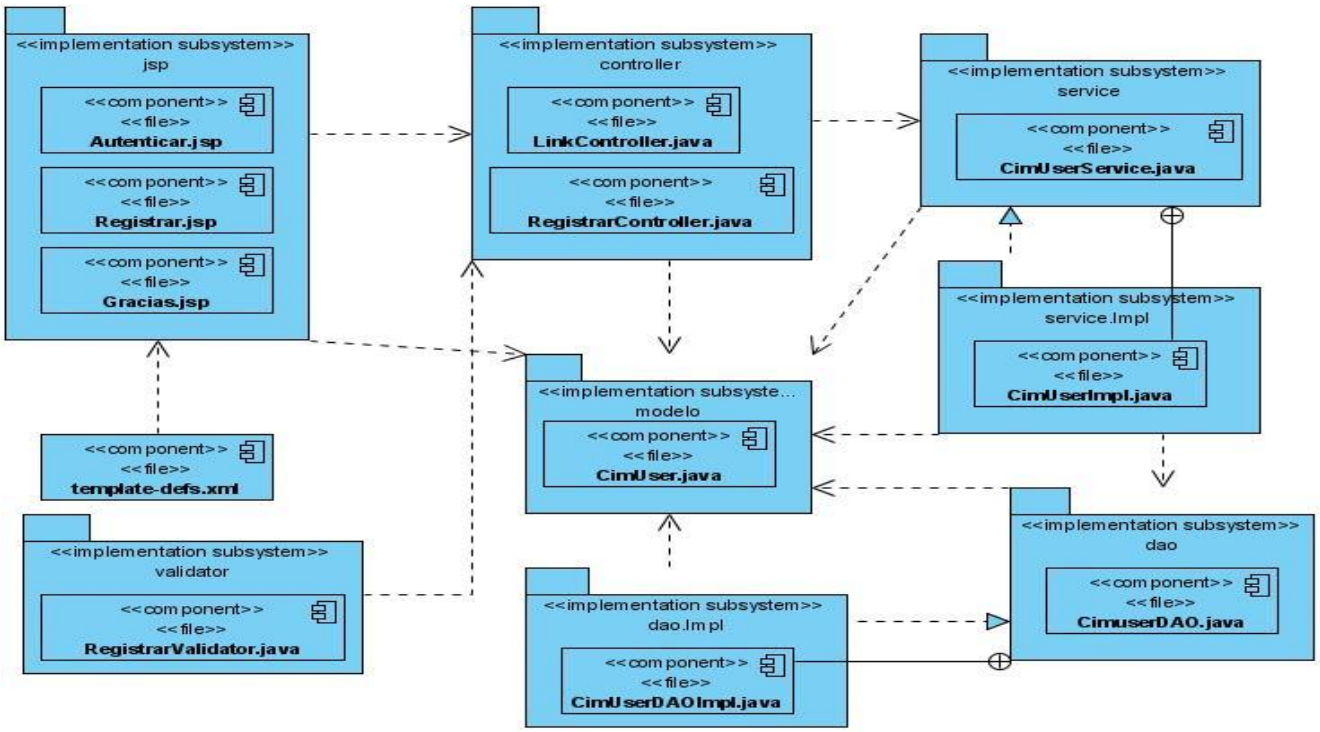
Anexo 23. Diagrama de secuencia CU Visualizar Proteína.



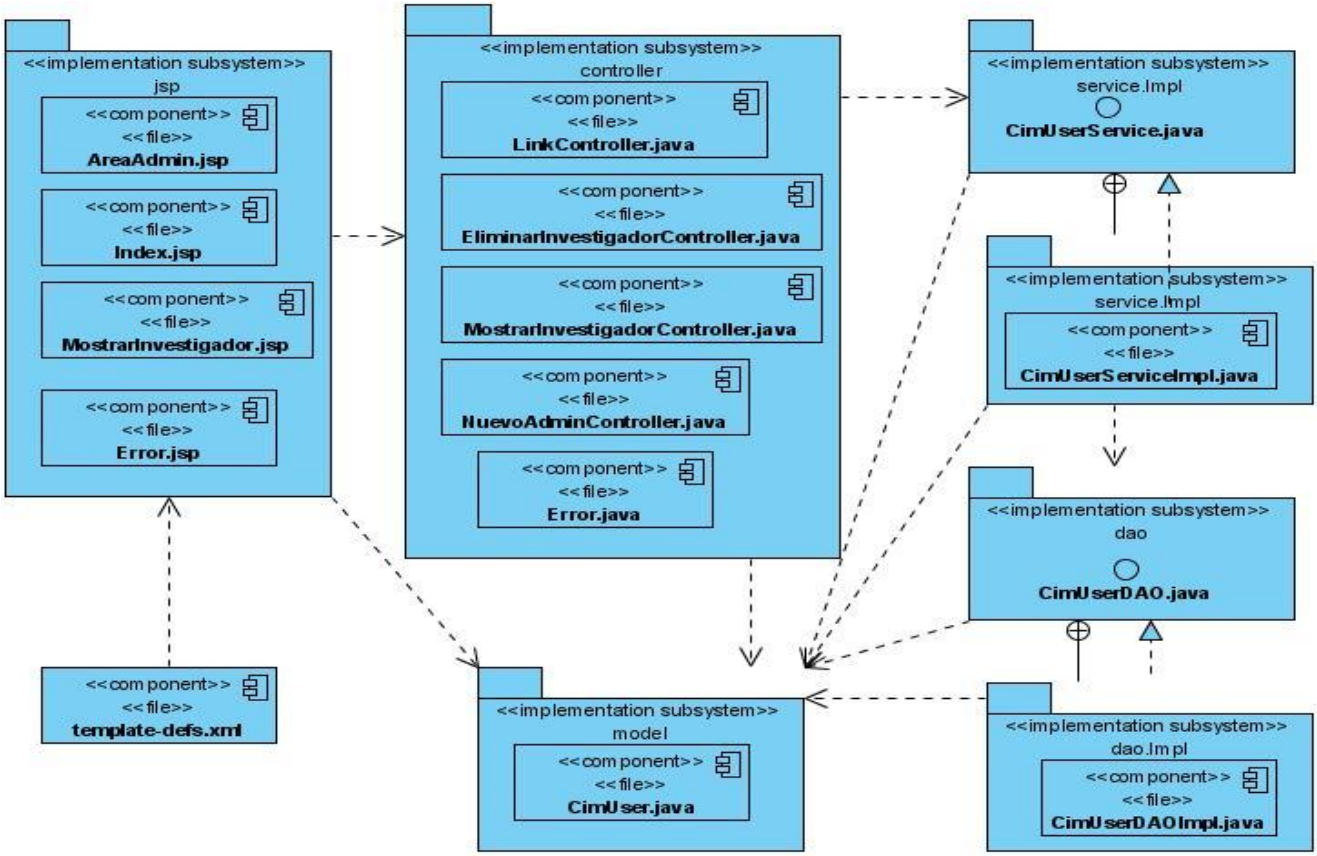
Anexo 24. Diagrama de componentes del Caso de Uso Autenticar Usuario.



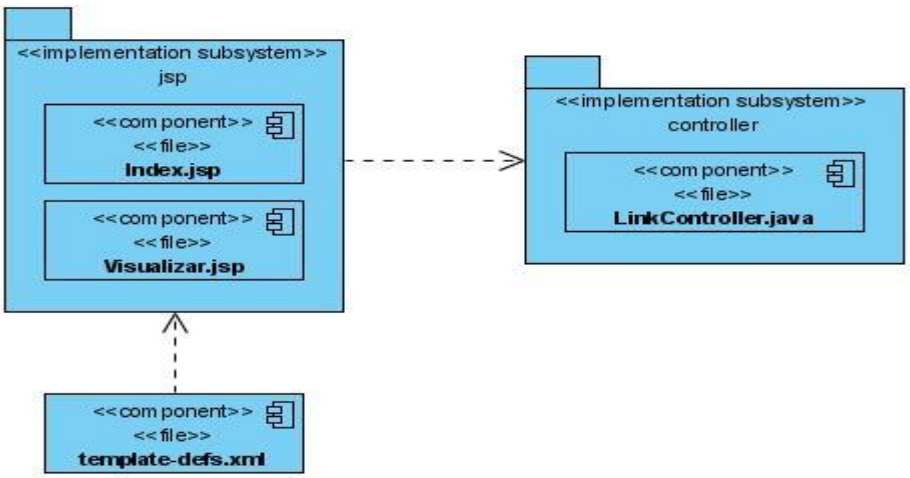
Anexo 25. Diagrama de componentes del Caso de Uso Registrar Investigador.



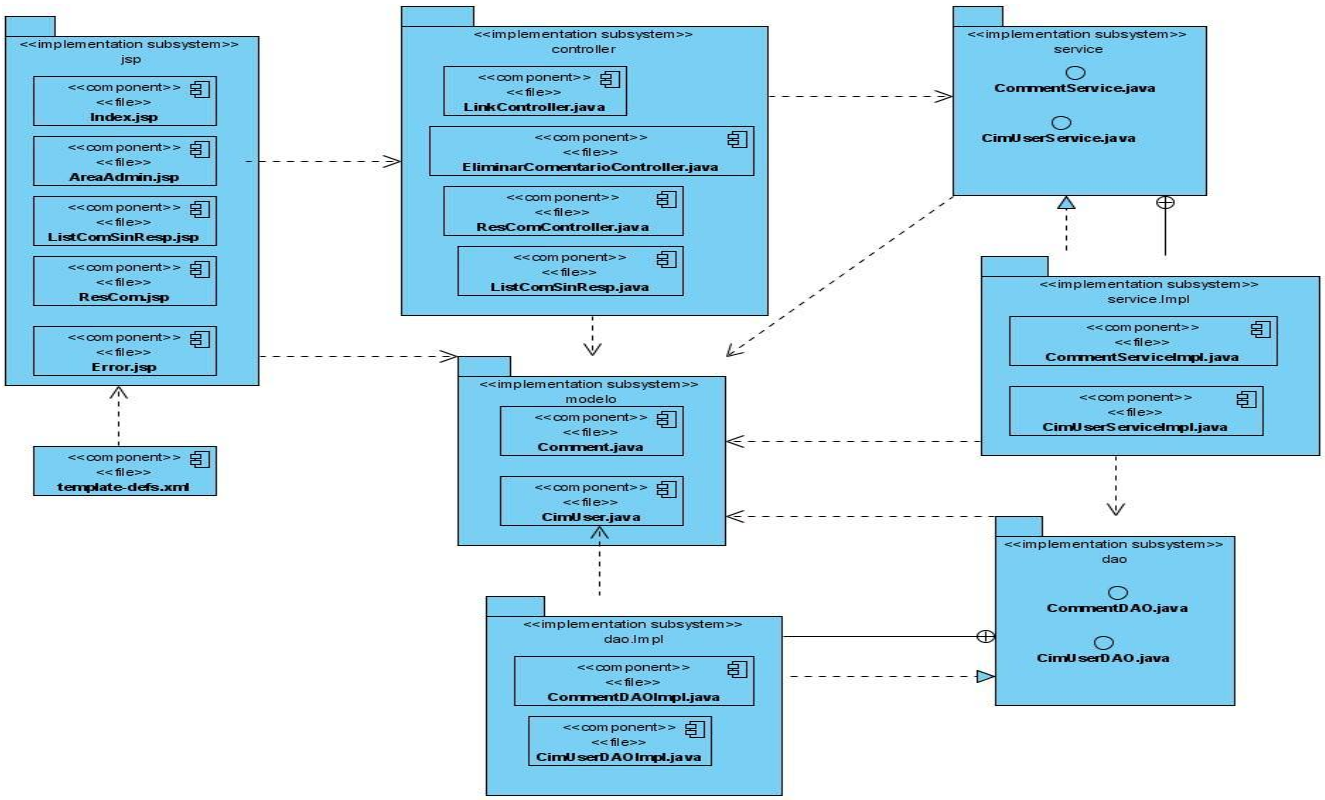
Anexo 26. Diagrama de componentes del Caso de Uso Gestionar Investigadores.



Anexo 25. Diagrama de componentes del Caso de Uso Visualizar Proteína.



Anexo 26. Diagrama de componentes del Caso de Uso Gestionar Comentarios del Administrador.



Anexo 28. Diagrama de componentes del Caso de Uso Gestionar Comentarios del Investigador.