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

Servicio de análisis automático de poses derivadas del docking utilizando AuPosSOM para la Plataforma de Servicios Bioinformáticos.

Autor: Yasmany Jesús Alonso Cabrera

Tutor: MsC. Alina Agramonte Delgado

Co-Tutor: Ing. Osvel Chávez Hernández

La Habana 2013

"Año 55 de la Revolución"

“El querer lo es todo en la vida. Si queréis ser felices lo seréis. Es la voluntad la que transporta las montañas.”

Alfred Victor de Vigny

Declaración de Autoría.

Yo Yasmany Jesús Alonso Cabrera declaro ser el autor del trabajo de diploma con título: Servicio de análisis automático de poses derivadas del docking utilizando AuPosSOM para la Plataforma de Servicios Bioinformáticos y admito los derechos patrimoniales del mismo con carácter exclusivo a la Universidad de las Ciencias Informáticas, específicamente a la Facultad 6.

Para que así conste, firmo el presente a los ____ días del mes de _____ del año 2013.

_____

Firma del autor

Yasmany Jesús Alonso Cabrera

_____ _____

Firma del tutor

Alina Agramonte Delgado

Firma del co-tutor

Osvel Chávez Hernández

Datos de Contacto.

Autor:

Yasmany Jesús Alonso Cabrera
Universidad de las Ciencias Informáticas
e-mail: yjalonso@estudiantes.uci.cu

Tutora:

Alina Agramonte Delgado
Licenciada en Química
Máster en Bioinformática
e-mail: alinaad@uci.cu

Co-Tutor:

Osvel Chávez Hernández
Ingeniero en Ciencias Informáticas
e-mail: ochavez@uci.cu

Agradecimientos.

Ante todo agradecer a mi familia por haberme apoyado y aguantado a lo largo de estos cinco años y durante toda mi vida, gracias por educarme y enseñarme valores que han constituido la base de mi desarrollo como persona, como estudiante y en lo adelante, como profesional.

A mis padres que son mis pilares fundamentales y a los cuales les dedico este logro, a mis abuelos por estar siempre pendientes de mí y ayudarme a superar cualquier dificultad, a mi tía por ser mi segunda madre y una guía para mí tanto en lo personal como en lo profesional.

A mi novia por apoyarme y tener paciencia conmigo durante este difícil año en que la conocí, por brindarme su amor pese a todas las dificultades y por creer en mí y no dejarme caer en malos pasos.

A mis tutores, los profesores Alina y Osvel por permitirme hacer este trabajo y guiarme a través del camino de la confección del mismo, por explicarme y aclararme todas las dudas que han nacido en una encrucijada como esta.

A mi tribunal y a mi oponente por hacerme las críticas acertadas y oportunas que me ayudaron a crecer como profesional y ayudarme a confeccionar un producto con la calidad requerida por una tesis para optar por el título de Ingeniero en Ciencias Informáticas.

A todos mis amigos y compañeros de los grupos 6105, 6205, 6309,6409 y 6509, a mis colegas de los apartamentos 75206, 94306, 110101 y 95102, aunque algunos ya no estén con nosotros en la Universidad pero igualmente contribuyeron a formarme como profesional y tantos buenos momentos que pasaron junto a mí.

A todos mis profesores que me educaron durante el transcurso de estos cinco años y me ayudaron a formar valores tanto profesional como personalmente.

En fin a todos los que me apoyaron y dedicaron un poquito de su tiempo para ayudarme a llegar donde estoy hoy.

El autor

Dedicatoria.

Dedicado a mi familia que ha sacrificado mucho porque yo llegara hasta aquí y en especial a mis padres por ayudarme a ser un poco más como ellos.

El autor

Resumen.

Antiguamente el desarrollo de medicamentos tuvo como base el conocimiento tradicional y el trabajo de laboratorio para el aislamiento de sus principios activos. En la actualidad se han creado métodos automatizados de simulación del acoplamiento molecular o docking, que contribuyen al diseño de fármacos con mayor potencia medicinal y menor toxicidad. En la Universidad se brinda este servicio mediante el módulo de Acoplamiento Molecular existente en la Plataforma de Servicios Bioinformáticos, sin embargo, no existe una herramienta capaz de interpretar los resultados arrojados por estos métodos de simulación en dicha plataforma. El presente estudio propone un Servicio de Análisis Automático de Poses Derivadas del Docking utilizando el programa AuPosSOM para insertar en el módulo de Acoplamiento Molecular de la Plataforma de Servicios Bioinformáticos de la UCI. Para ello se siguieron como principales pasos: la identificación de las herramientas de software más usadas para el desarrollo del nuevo servicio, el análisis y el diseño para insertar el AuPosSOM al módulo de Acoplamiento Molecular, y la implementación y evaluación del servicio de Análisis Automático de Poses Derivadas del Docking para la Plataforma de Servicios Bioinformáticos. El trabajo logró un servicio web de Análisis Automático de Poses Derivadas del Docking que contiene además una base de datos relacional para el almacenamiento de un gran volumen de datos. El servicio brindará al usuario la posibilidad de agrupar, a partir de los resultados del docking, las poses de salida de los compuestos en cluster por semejanza estructural, lo que facilita el análisis de los resultados y simplifica el trabajo al investigador.

Palabras Clave: Acoplamiento molecular, Docking, Poses Derivadas del Docking, AuPosSOM, servicio web.

Índice

Introducción.

Muchos medicamentos actualmente en uso deben su descubrimiento tanto al estudio de remedios tradicionales como al aislamiento de principios activos de productos naturales, o bien a cierto azar, como en el caso de la penicilina. Sin embargo, en el desarrollo moderno de fármacos se utilizan tecnologías sofisticadas para realizar búsquedas sistemáticas a gran escala con el propósito de encontrar sustancias cada vez con mayor potencia medicinal y menor toxicidad. La primera etapa consiste en la elección de la molécula blanco, sobre la que actuará el medicamento para lograr el efecto terapéutico deseado. Diversas investigaciones sobre los fármacos empleados en las últimas décadas han revelado que la acción terapéutica de éstos incide principalmente en proteínas [1], ya sea como receptores moleculares o como enzimas. Por esta razón, en el diseño racional de medicamentos las proteínas pueden considerarse como la primera opción para fungir como moléculas blanco.

El método de acoplamiento molecular o docking, es un método de simulación por computadora para el desarrollo de fármacos que ha rendido resultados exitosos. La técnica del docking representa una alternativa a los procedimientos de ensayo masivo de laboratorio, también, y de manera más importante, ella ha sido utilizada en el diseño de fármacos, tanto para llevar a cabo las etapas de identificación del sitio de unión sobre el blanco, como la de construcción y evaluación de los complejos moleculares resultantes. Durante los ensayos masivos que se llevan a cabo durante el docking, a veces es necesario probar bases de datos de cientos de ligandos contra un solo receptor, esto requiere de un gran despliegue de recursos puesto que es necesario obtener, procesar y almacenar cantidades colosales de resultados obtenidos in silico. Frecuentemente el resultado de este tipo de simulaciones genera una elevada cantidad de posible poses que puede adoptar el ligando que puntualmente se esté probando, quedando por la parte del investigador el identificar aquellas que tengan contactos importantes que puedan explicar la posible actividad biológica.

Esta forma de trabajo experimental ha resultado de gran valía para entender de qué manera los fármacos que utilizamos diariamente alcanzan su objetivo dentro de nuestro cuerpo, y también facilita el desarrollo de nuevos fármacos con características bien definidas o con objetivos seleccionados previamente.
En la Universidad de las Ciencias Informáticas (UCI) se cuenta con la Plataforma de Servicios Bioinformáticos donde existe un módulo dedicado al acoplamiento molecular que brinda este servicio a través del programa Dock6.5, utilizando la plataforma T-arenal como alternativa para el cálculo a gran escala usando estaciones de trabajos no dedicadas. Actualmente no existe una herramienta capaz de interpretar los resultados arrojados por el servicio de docking brindado por dicha plataforma. El presente trabajo centra su atención en el programa AuPosSOM porque, a través de mapas auto-organizativos, es capaz de identificar automáticamente importantes contactos para la actividad biológica entre el ligando y el blanco, facilitando la interpretación de los resultados relacionados con el análisis estructural.

De esta problemática se deriva el **problema a resolver** ¿Cómo analizar las poses de salida del docking en el módulo de Acoplamiento Molecular de la Plataforma de Servicios Bioinformáticos? La investigación tiene como **objeto de estudio** el análisis de poses automáticas derivadas del docking.

La presente investigación tiene como **objetivo general:** Desarrollar un servicio de análisis automático de poses derivadas del docking utilizando AuPosSOM para la Plataforma de Servicios Bioinformáticos de la UCI. Se tiene como **campo de acción** el programa AuPosSOM en el módulo de Acoplamiento Molecular de la Plataforma de Servicios Bioinformáticos. Siendo los **objetivos específicos** del trabajo:

1. Identificar y caracterizar las herramientas de software más usadas para el análisis automático de las poses derivadas del docking.
2. Realizar el análisis y el diseño para insertar el AuPosSOM al módulo de Acoplamiento Molecular de la Plataforma de Servicios Bioinformáticos de la UCI.

3. Implementar el servicio de Análisis Automático de Poses Derivadas del Docking usando AuPosSOM para la Plataforma de Servicios Bioinformáticos de la UCI.
4. Evaluar el nuevo servicio de Análisis Automático de Poses Derivadas del Docking usando AuPosSOM en el módulo de Acoplamiento Molecular de la Plataforma de Servicios Bioinformáticos de la UCI.

A los que se le dará cumplimiento mediante las siguientes **tareas de la investigación:**

1. Realización del diseño teórico de la investigación.
2. Generación de los artefactos correspondientes a las fases de análisis y diseño para la adición del AuPosSOM al módulo de Acoplamiento Molecular de la Plataforma de Servicios Bioinformáticos de la UCI.
3. Diseño de la Interfaz Web que se va a incluir en la Plataforma de Servicios Bioinformáticos de la UCI para que los especialistas interactúen con el nuevo servicio de Análisis Automático de Poses Derivadas del Docking.
4. Implementación de la Interfaz Web y los componentes de software necesarios para brindar acceso al nuevo servicio de Análisis Automático de Poses Derivadas del Docking.
5. Realización de los casos de prueba de caja negra para el nuevo servicio de Análisis Automático de Poses Derivadas del Docking con AuPosSOM, de la Plataforma de Servicios Bioinformáticos de la UCI.
6. Ejecución de los casos de prueba para evaluar el funcionamiento del AuPosSOM en el módulo de Acoplamiento Molecular.

Teniendo como **posibles resultados:**

La Plataforma de Servicios Bioinformáticos de la UCI contará con un módulo de Servicios de Análisis Automático de Poses Derivadas del Docking que brindará al usuario la posibilidad de agrupar, a partir de los resultados del docking, las poses de salida de los compuestos en cluster por semejanza estructural.

Capítulo 1: Fundamentación Teórica.

En este capítulo se hará una búsqueda y comparación de las herramientas para el análisis de los resultados del Acoplamiento Molecular o Docking, así como las herramientas para el montaje de un Servicio de Análisis Automático de Poses Derivadas del Docking y las herramientas, metodologías, lenguajes y tecnologías para la construcción de la aplicación.

1.1 Portal de Servicios Bioinformáticos de la UCI.

Es un portal desarrollado sobre el contenedor de Portlets, Liferay Portal [1], que provee un ambiente amigable para la definición y la ejecución de análisis en la esfera de la bioinformática. Su objetivo es proporcionarle a los especialistas bioinformáticos el acceso a recursos computacionales, los cuales muchas veces carecen en sus centros de trabajo o de estudios [2].

1.2 Docking o Acoplamiento Molecular.

El acoplamiento molecular o docking es un método computacional que busca formas de unión entre ligandos potenciales (el fármaco a probar) y un blanco macromolecular, cuya estructura es conocida experimentalmente. Particularmente, el acoplamiento molecular se aplica para encontrar la orientación y posición de un ligando en el sitio activo de su blanco macromolecular, sin conocer el resultado final, es decir, la conformación tridimensional de la unión ligando-receptor. Esta última es normalmente estudiada por la cristalografía de rayos X, o analizando los complejos de energía de la resonancia magnética nuclear.

El objetivo de la técnica consiste en encontrar la pose de unión más probable entre el ligando y el receptor, es decir, la que menos energía requiera (a menor energía, más probable la unión) así como el sitio idóneo de unión molecular. La selección de la proteína específica de trabajo dependerá de la información disponible sobre el mecanismo molecular responsable del desarrollo de la enfermedad particular que se estudie [3]. Al conocer esta respuesta, el investigador está en posición de llevarla a la experimentación en vivo y así

corroborar sus resultados obtenidos in silico [4]. A continuación se explican brevemente algunas de las herramientas de docking más usadas en la actualidad.

1.2.1 AutoDock Vina.

AutoDock Vina es un programa de código abierto para el descubrimiento de nuevos medicamentos, acoplamiento molecular y cribado virtual, ofrece capacidad multi-núcleo, alto rendimiento, precisión mejorada y facilidad de uso.

Para los archivos de entrada y salida, el Vina usa el mismo archivo de estructura molecular PDBQT (Protein Data Bank, Partial Charge (Q), & Atom Type (T) por sus siglas en inglés) usado en AutoDock. Los archivos PDBQT pueden ser generados interactivamente o en modo consola y pueden ser visualizados usando un visualizador molecular [5].

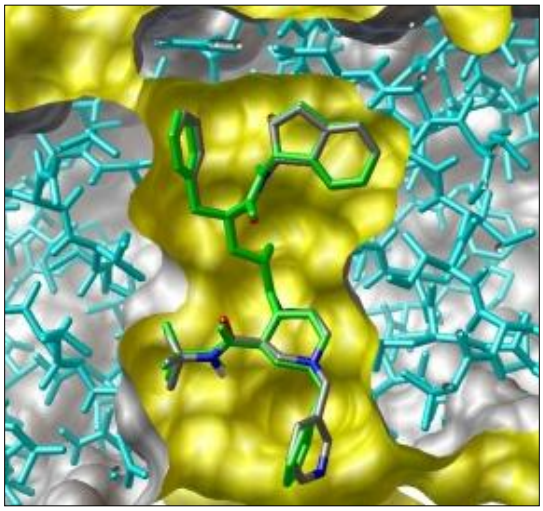


Fig. 1. Ejemplo de salida del AutoDock Vina de docking flexible superpuesto en las estructuras de cristal de indinavir (Imagen tomada de http://vina.scripps.edu).

1.2.2 Dock.

Es una herramienta de código abierto, escrita en C++ que realiza acoplamiento molecular [6]. Utiliza un modelo 3D de la proteína objetivo y gira el ligando sobre el área elegida de dicha proteína para lograr una mejor conformación entre ella y el ligando [7]. Al ser compilado el código fuente para una arquitectura de grid, hace sus corridas de forma paralela sobre el cluster donde esté instalado.

1.3 Análisis de los resultados del Docking.

Estas herramientas tienen en común que todas realizan acoplamiento molecular. Algunos simplifican el uso de los ficheros de entrada como es el caso del Vina con respecto al Dock y al AutoDock facilitándole así el trabajo al investigador. Pero ninguno de estos programas realiza un análisis de los resultados obtenidos, es decir no analizan las poses de los complejos ligando receptor-resultantes, quedando por parte de los investigadores el seleccionar aquellas poses que más se ajusten según la energía reportada por el docking.

Los programas de docking dan un listado ordenado por energía de asociación de cada uno de los complejos de salida siendo el de menor energía la opción más viable. Pero los investigadores no pueden conformarse con esto y tienen que analizar estructura por estructura. El AuPosSOM clusteriza estos resultados por criterios estructurales y muestra gráficamente las conclusiones, facilitando el trabajo al investigador. Evita la instalación y practica con varios softwares para analizar las salidas, además es para usuarios especialistas pero no necesariamente informáticos.

1.3.1 AuPosSOM.

Es una herramienta de cribado virtual para el análisis automático de estructuras moleculares acopladas. El análisis de contactos toma en cuenta puentes de hidrógeno, cargas electrostáticas, hidrofobia y todos los contactos entre ligandos candidatos a fármacos y proteínas. Desarrolla una función para identificar compuestos activos en un árbol. Una red neural permite la clasificación por homología de los ligandos en los contactos hechos con el blanco.

A partir de archivos de estructuras en formato PDB (Protein Data Bank por sus siglas en inglés), AuPosSOM determina contactos entre la molécula y el blanco. Un vector es construido representando la huella de la molécula en el blanco. Este vector es el equivalente algebraico de “n” poses del docking, donde “n” puede ser 1 [8]. La clasificación de los vectores homólogos se hace a través de una red neural Kohonen auto-organizativa, los archivos de salida generados tienen formato NEWICK y representan el resultado del cribado en forma de árbol, donde las hojas son los ligandos conocidos agrupados y organizados por categorías y la longitud

de las ramas está relacionada con la diferencia entre las huellas de las moléculas [9].

1.3.2 FigTree.

FigTree es un visualizador gráfico de árboles filogenéticos y un programa para producir imágenes. Está escrito en Java y utiliza, además de tener una interfaz gráfica amigable; una serie de códigos muy sencillos por consola para obtener gráficos a partir de archivos TREE en varios formatos como GIF, JPEG y PDF. En la figura 2 se muestra un ejemplo de salida del AuPosSOM visualizado con el FigTree.

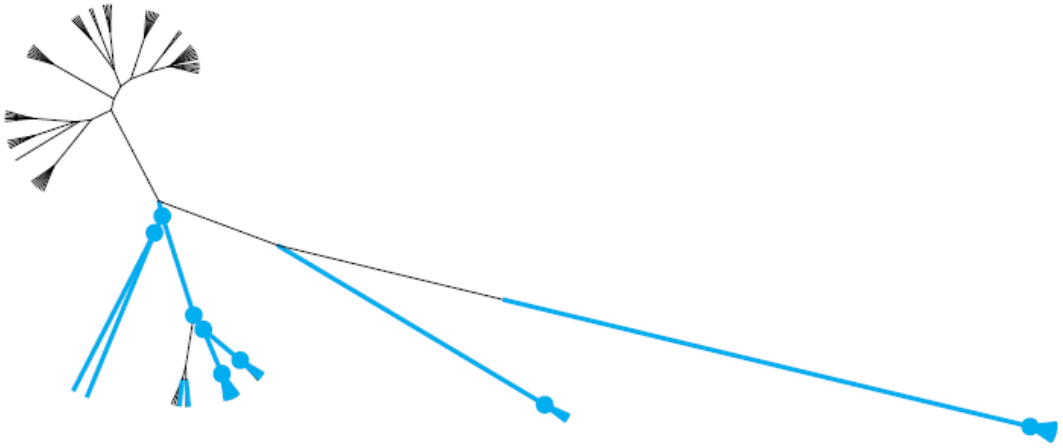


Fig. 2. Ejemplo de salida del AuPosSOM en forma de árbol visualizado con el FigTree, donde las hojas son los ligandos conocidos y la longitud de las ramas está relacionada con la diferencia entre las huellas de las moléculas (Imagen tomada de “AuPosSOM : Automatic analysis of Poses using Self-Organizing Map” [8]).

1.4 Tecnologías Web.

1.4.1 Apache Tomcat.

Es un software de código abierto que implementa tecnologías Java Servlet y Java Server Pages [10]. En él se pueden montar aplicaciones de misión crítica a través de la red. Es un servidor HTTP y un contenedor de servlets, cargándolos y ejecutándolos de forma dinámica.

1.4.2 Portlet.

Es un componente Web basado en la tecnología Java que procesa los pedidos del usuario y genera un contenido dinámico final [11], el contenido generado por un Portlet es llamado fragmento y su ciclo de vida es manejado por el contenedor de Portlets. Puede declarar los eventos que quiere emitir y los que desea recibir. El contenedor de Portlet actuará como intermediario y distribuirá los eventos en consecuencia.

1.4.3 Liferay Portal.

Liferay Portal es una plataforma web corporativa que permite desarrollar soluciones empresariales con resultados inmediatos y valor a largo plazo [12]. Es un portal de gestión de Portlets, de código abierto e implementado en Java. Se puede desplegar en servidores como Apache Tomcat y es multiplataforma lo que le permite ejecutarse en cualquier sistema operativo.

1.5 Marcos de trabajo.

1.5.1 Spring.

Un framework liviano que tiene como principales características la técnica Inversión de Control (IoC por sus siglas en inglés) y una implementación de desarrollo según el paradigma de Orientación a Aspectos (AOP por sus siglas en inglés). Define la forma de desarrollar aplicaciones J2EE, dando soporte y simplificando complejidad propia del software corporativo. La Inversión de Control promueve el bajo acoplamiento a partir de la inyección de dependencias (DI por sus siglas en inglés) entre los objetos (relaciones) y la Orientación a Aspectos presenta una estructura simplificada para el desarrollo y utilización de aspectos (módulos multiple object cross cutting) [13]. Un elemento clave de Spring es el soporte infraestructural a nivel de aplicación: Spring se enfoca en la base de las aplicaciones corporativas y así los equipos pueden enfocarse en las aplicaciones a nivel de negocio [14].

1.5.2 Axis2.

Es un middleware, que permite la comunicación con otras aplicaciones, que reestructuró la arquitectura del módulo Apache Web Service para incorporar los cambios sobre Servicios Web. Su arquitectura se basa en desarrollar sobre un núcleo simple y extensible que proporciona las abstracciones básicas para el resto del sistema [15].

1.5.3 Hibernate.

Hibernate es una herramienta de mapeo objeto/relacional para entornos Java. El término de mapeo objeto/relacional (ORM por sus siglas en inglés) se refiere a la técnica de mapear una representación de datos desde un modelo de objeto a un modelo de datos relacionales con un esquema basado en SQL [16]. Permite al desarrollador detallar cómo es su modelo de datos, qué relaciones existen y qué forma tienen. Genera las sentencias SQL y libera al desarrollador del manejo manual de los datos que resultan de la ejecución de dichas sentencias, manteniendo la portabilidad entre todos los motores de bases de datos con un ligero incremento en el tiempo de ejecución.

1.6 Lenguaje de Programación.

Un lenguaje de programación es un lenguaje diseñado para describir el conjunto de acciones consecutivas que un equipo debe ejecutar. Por lo tanto, un lenguaje de programación es un modo práctico para que los seres humanos puedan dar instrucciones a un equipo [17].

1.6.1 Java.

Es un lenguaje orientado a objetos desarrollado bajo una licencia libre. Fue desarrollado por la Sun Microsystems con la idea de utilizarse para el desarrollo de aplicaciones web, aunque luego se percataron de la amplia gama de posibilidades que tiene sobre las aplicaciones de escritorio. Los programas desarrollados con Java son independientes de la arquitectura de la computadora donde se ejecutan, también son independientes del sistema operativo que los

ejecuta ya que el código Java no se ejecuta sobre el sistema operativo sino sobre una máquina virtual (Java Virtual Machine, JVM por sus siglas en inglés) [17].

1.7 Entorno de Desarrollo Integrado.

Un entorno de desarrollo integrado (IDE) (también conocida como entorno de diseño integrado, entorno integrado de depuración o entorno de desarrollo interactivo) es una aplicación de software que proporciona servicios integrales a los programadores de computadoras para el desarrollo de software. Una IDE normalmente se compone de un editor de código fuente, un compilador y / o un intérprete, generación automatizada de herramientas y un depurador [18].

1.7.1 Eclipse.

Desarrollado por IBM alrededor del año 2001 y cedido a la Fundación Eclipse sin fines lucrativos, para manejarlo como una plataforma de código abierto. Además de ser un IDE, Eclipse implementa numerosas funciones que de otra forma los usuarios tendrían que hacer a mano. Es fácil de usar e integra sus funciones unas con otras fácilmente [19].

1.8 Metodología de desarrollo de software.

Una metodología de desarrollo de software se refiere al entorno que se usa para estructurar, planificar y documentar el proceso de desarrollo de software.

1.8.1 OpenUP.

Es un proceso de desarrollo de software mínimamente suficiente, es decir, que solamente el contenido fundamental es incluido. Es una metodología ágil, por esto no provee una buena opción para muchos tópicos dentro de un proyecto, tales como amplios equipos de trabajo, conformidad, seguridad o aplicaciones críticas, etc. OpenUP tiene las características esenciales de Unified Process que aplica enfoques iterativo e incremental dentro de un ciclo de vida estructurado. Es basado en casos de uso, manejo de riesgos y tiene una arquitectura centralizada para guiar el desarrollo de software [20].

1.9 Lenguaje de modelado.

Un lenguaje de modelado es un conjunto de símbolos estandarizados representados de cierta manera para modelar parte de un diseño de software.

1.9.1 Unified Modeling Language.

El Lenguaje Unificado de Modelado (UML por sus siglas en inglés) prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan. Mientras que ha habido muchas notaciones y métodos usados para el diseño orientado a objetos, ahora los modeladores sólo tienen que aprender una única notación [21]. UML se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real.

1.10 Herramienta de Modelado.

CASE es un acrónimo para Computer-Aided Software Engineering. Esencialmente, una herramienta CASE ayuda al ingeniero de software a desarrollar y mantener software, son herramientas individuales para ayudar al desarrollador de software o administrador de proyecto durante una o más fases del desarrollo de software (o mantenimiento) [22].

1.10.1 Visual Paradigm.

Visual Paradigm para UML es una herramienta para desarrollo de aplicaciones utilizando modelado UML ideal para Ingenieros de Software, Analistas de Sistemas y Arquitectos de sistemas que están interesados en construcción de sistemas a gran escala y necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos [23]. Oferta una amplia gama de herramientas para el desarrollo de software, los equipos de desarrollo lo necesitan para la captura de requisitos, planeamiento de software, casos de prueba, modelado de datos y de clases, etc. [24].

1.11 Sistema gestor de base de datos.

Un sistema gestor de bases de datos o SGBD (aunque se suele utilizar más a menudo las siglas DBMS procedentes del inglés, Data Base Management System)

es el software que permite a los usuarios procesar, describir, administrar y recuperar los datos almacenados en una base de datos [25]. Proporcionan una visión abstracta de los datos. Utiliza una variedad de técnicas eficientes para almacenar y recuperar los datos así como su seguridad en el almacenamiento y transporte [26].

1.11.1 PostgreSQL.

Sistema de gestión de bases de datos objetos - relacional, de propósito general, multiplataforma, multiusuario, de código abierto y diseñado para ambientes de misión crítica y mantenimiento de grandes bases de datos. Soporta el almacenamiento de objetos binarios grandes. Contiene columnas auto-incrementales. Implementa la herencia de tablas y los disparadores comunes, por columna, condicionales. Permite las consultas recursivas [27].

1.12 Conclusiones del capítulo.

1. Según lo expuesto anteriormente para dar solución al problema planteado, se realizará un servicio web sobre el middleware Axis2 el cual se desplegará sobre un servidor Apache Tomcat 6.0.
2. Para el desarrollo del cliente que utilizará el servicio se empleará la metodología OpenUP por ser una metodología ágil y la utilizada en el proyecto Plataforma de Servicios Bioinformáticos.
3. Como herramienta de modelado se empleará el Visual Paradigm UML 6.4 por ser una herramienta que cubre todo el ciclo de vida del proceso de desarrollo de software y además no es privativa.
4. Como IDE de desarrollo se hará uso del Eclipse Helios 3.6, el cual permite trabajar con el lenguaje de programación Java que es el lenguaje propuesto por ser multiplataforma, orientado a objetos y de licencia libre al igual que el IDE escogido.
5. Como sistema gestor de base de datos se utilizará el PostgreSQL 8.4 ya que permite el almacenado de grandes extensiones de datos y presenta una licencia libre.

6. Como marco de trabajo se utilizarán el Spring 2.6 para la implementación de la Interfaz Web por ser de software libre y de código abierto, y el Hibernate 3.3.1 como herramienta de mapeo objeto-relacional por ser de software libre y distribuido bajo los términos de la licencia GNU/LGPL (GNU es un acrónimo recursivo para No es UNIX y LGPL es un acrónimo para Library General Public License por sus siglas en inglés).
7. Se utilizará el programa AuPosSOM para el análisis automático de las poses de salida del servicio de Acoplamiento Molecular de la Plataforma de Servicios Bioinformáticos.

Capítulo 2: Análisis y diseño.

En el presente capítulo se hace el análisis y diseño a la propuesta del sistema. En el análisis se especifican los requerimientos funcionales y no funcionales, las descripciones de los casos de uso y los actores así como la relación entre ellos mediante el diagrama de casos de uso del sistema. En el diseño se generan los artefactos correspondientes a esta etapa, se describe la arquitectura y se definen los patrones de diseño, se describen los diagramas de clases del diseño y el modelo de despliegue.

2.1 Modelo de Dominio.

Permite definir el alcance de un sistema, quiénes y qué elementos están involucrados. Representa un modelo conceptual de las entidades que participan del negocio y sus relaciones [28]. El mismo define un modelo de clases común para todos los involucrados en el modelo de requisitos [29]. El Modelo de Dominio del sistema se representa a continuación.

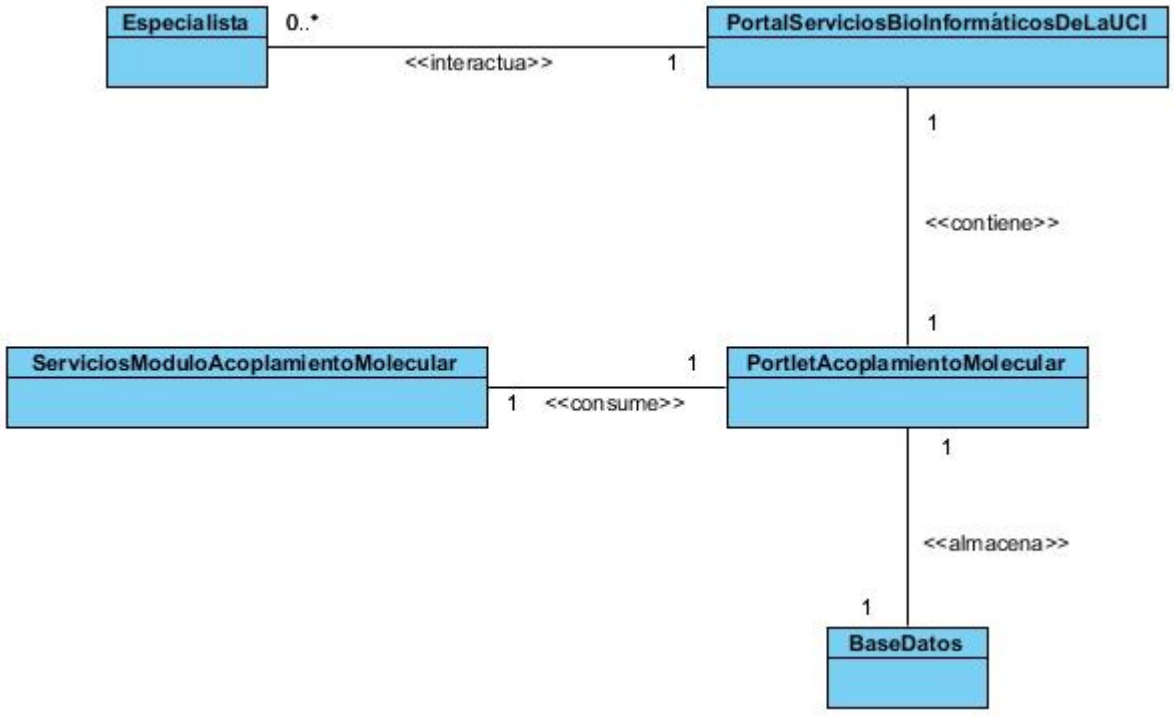


Fig. 3. Diagrama de Modelo de Dominio. Representa los conceptos del dominio del mundo real y las relaciones entre ellos.

2.1.1 Definición de las clases del Modelo de Dominio.

Especialista: Usuario que posee privilegios para interactuar con el Portal de Servicios Bioinformáticos de la UCI.

PortalDeServiciosBioInformáticosDeLaUCI: Portal con interfaces gráficas donde los especialistas pueden consumir los servicios que brinda la Plataforma de Servicios Bioinformáticos de la UCI.

PortletAcoplamientoMolecular: Portlet que le permite al especialista la gestión de sus corridas y la administración de sus ejecuciones.

ServiciosModuloAcoplamientoMolecular: Es donde se publican todos los servicios pertenecientes al módulo de Acoplamiento Molecular.

BaseDatos: Es una base de datos relacional donde se almacenan los datos de los procesos que serán ejecutados en el Portlet de Acoplamiento Molecular.

2.2 Requisitos del sistema.

Los requisitos son las condiciones que el sistema debe alcanzar o cumplir. A continuación se presentan los requisitos separados en funcionales y no funcionales.

2.2.1 Requisitos funcionales.

Los requisitos funcionales son el resultado de la entrevista con el cliente y definen el comportamiento interno del sistema. A continuación se enumeran.

RF1 Realizar Análisis Automático de Poses Derivadas del Docking: Este sistema deberá ser capaz de realizar Análisis Automático de Poses Derivadas del Docking.
RF2 Modificar proceso de Análisis Automático de Poses Derivadas del Docking: Mediante este requisito el sistema deberá brindar la posibilidad de modificar el proceso de Análisis Automático de Poses Derivadas del Docking.

RF3 Listar proceso de Análisis Automático de Poses Derivadas del Docking: Este sistema deberá brindar la posibilidad de listar los procesos de Análisis Automático de Poses Derivadas del Docking de un especialista.

RF4 Eliminar proceso de Análisis Automático de Poses Derivadas del Docking: A través de este requisito, el sistema deberá brindar la oportunidad de eliminar los procesos de Análisis Automático de Poses Derivadas del Docking de un especialista.

RF5 Listar ficheros de salida del proceso de Análisis Automático de Poses Derivadas del Docking: Este sistema deberá brindar la posibilidad de visualizar los ficheros de salida del proceso de Análisis Automático de Poses Derivadas del Docking.

RF6 Visualizar ficheros de salida del proceso de Análisis Automático de Poses Derivadas del Docking: Este sistema deberá ser capaz de visualizar los ficheros de salida del proceso de Análisis Automático de Poses Derivadas del Docking.

RF7 Descargar ficheros de salida del proceso de Análisis Automático de Poses Derivadas del Docking: Este sistema deberá brindar la funcionalidad de descargar los ficheros de salida del proceso de Análisis Automático de Poses Derivadas del Docking.

2.2.2 Requisitos no funcionales.

Son propiedades o cualidades que el producto debe tener, son requisitos que imponen restricciones en el diseño o en la implementación. Son características que hacen al producto atractivo, usable, rápido o confiable. A continuación se muestran.

Seguridad.

Los especialistas tienen que estar autenticados en el Portal de Servicios Bioinformáticos de la UCI para acceder a todas las funcionalidades del sistema. La aplicación contará con protección contra acciones no autorizadas y que puedan afectar la integridad de los datos por lo que cada especialista solo tendrá acceso a su propia información, garantizando así la confidencialidad.

Rendimiento.

El sistema está concebido para un ambiente cliente - servidor por lo que la rapidez del servicio dependerá del volumen de la información que necesite procesar y de la cantidad de corridas en la cola de ejecución.

Usabilidad.

El sistema se desplegará en el Portal de Servicios Bioinformáticos de la UCI. Tendrá un ambiente sencillo y será fácil de manejar por cualquier investigador aunque no tenga mucha experiencia en el trabajo con las computadoras. Contendrá un manual de usuario con las descripciones básicas de cómo trabajar con el sistema. Deberá estar disponible las 24 horas del día los 7 días de la semana garantizando un acceso de forma fácil y rápida para los especialistas.

Apariencia o interfaz externa.

El sistema debe contar con una interfaz amigable que permita al especialista interactuar de forma cómoda, le agilice y facilite el trabajo con el sistema. La página principal mostrará la guía de uso que servirá de ayuda al especialista.

Portabilidad.

El sistema podrá ser ejecutado en cualquier sistema operativo, por su característica de ser multiplataforma, permitiendo una fácil migración haciendo uso de estándares y tecnologías de código abierto.

Soporte.

Una vez terminado el sistema se agregará al Portal de Servicios Bioinformáticos de la UCI para realizar las pruebas piloto y de despliegue, luego quedará instalada en dicha plataforma para su uso. Se actualizará a medida que se diseñen las mejoras.

Software.

En el cliente: Se debe disponer para ejecutar la aplicación:

- Navegador Web estándar con capacidad de interpretación de Java Script, CSS (Cascade Style Sheet por sus siglas en inglés) compatible con la W3C

(acrónimo para World Wide Web Consortium), optimizado para Mozilla FireFox 9.0 o superior.

En el servidor: Se debe disponer para poder instalar la aplicación:

- Servidor Web Apache Tomcat 6.0.14 o superior.
- JDK6 o superior.
- PostgreSQL 8.4 o superior.

Hardware.

En el cliente: Se debe disponer para ejecutar la aplicación:

Para las estaciones de trabajo de los especialistas se necesita una PC con microprocesador Intel Pentium IV o superior, con un mínimo de 512MB de memoria RAM y tarjeta de red para la conexión de red con el Portal de Servicios Bioinformáticos de la UCI.

En el servidor: Se debe disponer para poder instalar la aplicación.

Los servidores deberán contar con un microprocesador Intel Pentium IV o superior, con un mínimo de 1GB de memoria RAM, un disco duro con capacidad disponible de almacenamiento de 40GB y una tarjeta de red para poder brindar el servicio utilizando la red.

Legales.

Se usarán herramientas de software libre y código abierto, o que funcionen bajo las licencias GNU/LGPL, por lo que el servicio que se brindará está basado también en la licencia GNU/LGPL.

2.3 Definición de los actores del Sistema.

Los actores representan quienes interactúan o hacen uso del sistema. Cada actor juega un rol dentro del sistema al interactuar con él, diferentes especialistas pueden asumir el mismo rol de actor.

Especialista:

Es el rol que podrá realizar un conjunto de tareas para hacer una corrida sobre el Servicio de Análisis Automático de Poses Derivadas del Docking y acceder a la respuesta dada por el mismo.

2.4 Definición de los Casos de Uso del Sistema.

Los casos de uso representan uno o varios requisitos funcionales, por lo que representa funcionalidades que el sistema ofrece para que los actores puedan interactuar con ellos. A continuación se listan.

CU1 Realizar Análisis Automático de Poses Derivadas del Docking.
RF1 Realizar Análisis Automático de Poses Derivadas del Docking.
CU2 Administrar proceso de Análisis Automático de Poses Derivadas del Docking.
RF2 Modificar proceso de Análisis Automático de Poses Derivadas del Docking.
RF3 Listar proceso de Análisis Automático de Poses Derivadas del Docking.
RF4 Eliminar proceso de Análisis Automático de Poses Derivadas del Docking.
CU3 Listar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.
RF5 Listar ficheros de salida del proceso de Análisis Automático de Poses Derivadas del Docking.
CU4 Visualizar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.
RF6 Visualizar ficheros de salida del proceso de Análisis Automático de Poses Derivadas del Docking.
CU5 Descargar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.
RF7 Descargar ficheros de salida del proceso de Análisis Automático de Poses Derivadas del Docking.

2.5 Realización de los Casos de Uso del Sistema.

Los diagramas de casos de uso documentan el comportamiento de un sistema desde el punto de vista del usuario. Por lo tanto los casos de uso determinan los

requisitos funcionales del sistema, es decir, representan las funciones que un sistema puede ejecutar [30]. A continuación se muestra el Diagrama de Casos de Uso del Sistema.

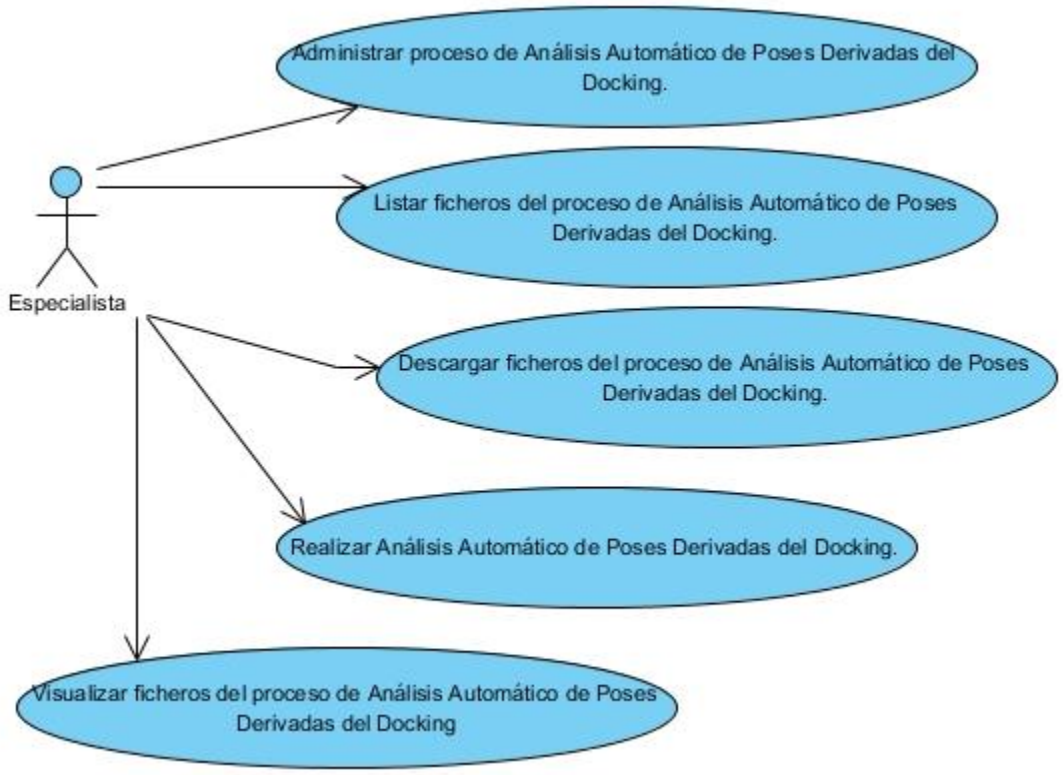


Fig. 4. Diagrama de Casos de Uso del Sistema, cada Caso de Uso puede agrupar varios Requisitos Funcionales y el Especialista es quien inicializa todos los Casos de Uso.

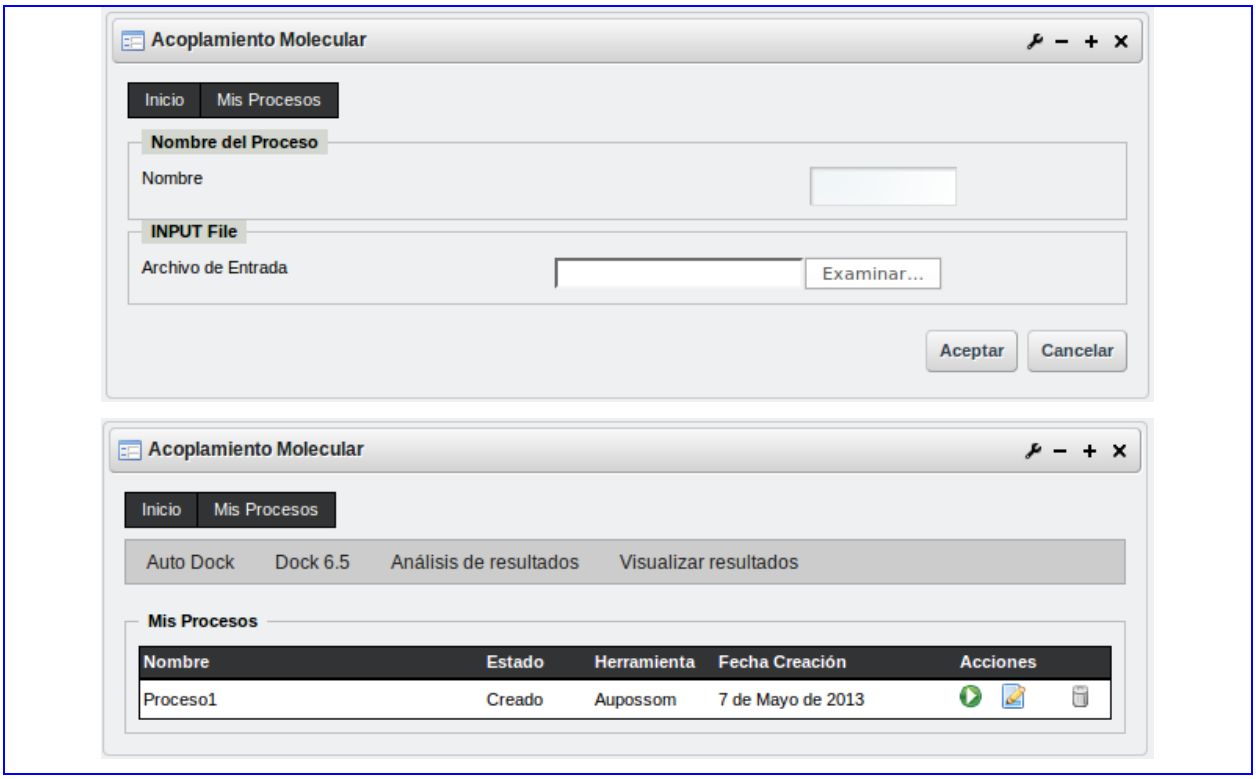
2.5.1 Descripción de los Casos de Uso del Sistema.

La descripción de los casos de uso del sistema narra detalladamente los eventos que los actores realizan para completar un proceso o funcionalidad a través de la aplicación. A continuación se describe minuciosamente los Casos de Uso presentes en el Diagrama de Casos de Uso del Sistema.

Descripción textual del CU Realizar Análisis Automático de Poses Derivadas del Docking.

Caso de Uso:	Realizar Análisis Automático de Poses Derivadas del Docking.
Actores:	Especialista
Resumen:	El caso de uso comienza cuando el especialista necesita analizar las poses de salida del Docking. El sistema le brinda un formulario en el cual se sube los archivos de entrada para realizar el análisis. El caso de uso

	finaliza cuando se genera el archivo de salida en forma de árbol.	
Precondiciones:		
Referencias	RF1.	
Prioridad	Crítica.	
Flujo Normal de Eventos		
Sección "Realizar Análisis Automático de Poses Derivadas del Docking"		
Acción del Actor		**Respuesta del Sistema**
1.	El especialista inicia el caso de uso cuando selecciona la opción "Análisis de resultados"-->"AuPosSOM" en la interfaz principal.	
2.		Muestra en la interfaz un formulario con los siguientes campos: • Nombre del proceso. • Archivos de entrada.
3.	Introduce los datos necesarios (nombre del proceso y archivos de entrada) para la creación del proceso y presiona el botón "Aceptar".	
4.		El sistema valida los datos entrados y si son correctos crea el nuevo proceso en la base de datos.
5.	El usuario presiona "Ejecutar" en el nuevo proceso creado.	
6.		El sistema analiza las poses y genera un archivo en forma de árbol.
7.		Fin del proceso.
Flujos Alternos		
		3.1. Si los datos entrados son incorrectos, el sistema muestra un mensaje mostrando el error en la interfaz.
Prototipo de Interfaz		



El resto de las descripciones de los CU identificados se encuentran en el anexo del trabajo.

2.5.2 Patrones arquitectónicos.

Modelo Vista Controlador (MVC).

El patrón MVC es una arquitectura de diseño software para separar los componentes de aplicación en tres niveles, interfaz de usuario, lógica de control y lógica de negocio. Es una especialización de un modelo de capas, con la diferencia que se usa para entornos web como patrón por excelencia [31]. A continuación se describe cómo se utilizan las capas en el sistema.

- La capa de presentación o vista está formada por las clases JSPs, que son las clases con las que interactúa directamente el especialista desde la Plataforma de Servicios Bioinformáticos de la UCI.
- La capa de control se implementa con una o varias Clases Controladoras que son las clases encargadas de recibir las diferentes peticiones de los especialistas y hacer las llamadas a las funciones necesarias para dar cumplimientos a estas peticiones.

- La capa de datos puede ser implementada a través de componentes que representan objetos en la base de datos, en nuestro caso Plain Old Java Object (POJO por sus siglas en inglés) con Hibernate. La base de datos almacena de forma persistente toda la información necesaria para facilitar el servicio de Acoplamiento Molecular en la Plataforma de Servicios Bioinformáticos de la UCI.

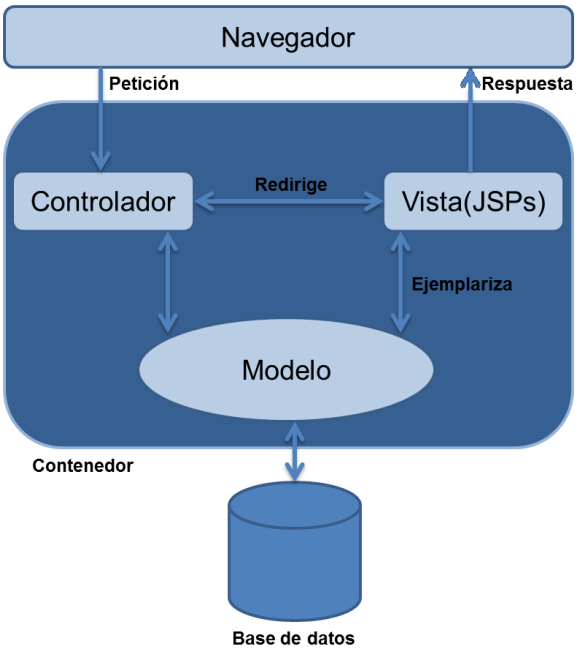


Fig. 5. Modelo Vista Controlador. La utilización del patrón MVC aumenta el desacoplamiento de los componentes del sistema en el que cada capa responde a la llamada de otra manteniendo la cohesión entre ellas [1].

Arquitectura del Portlet a implementar

El sistema se desarrolla bajo el patrón de arquitectura MVC para el desarrollo del Portlet. Contiene un objeto para el control, parametrizable desde un archivo XML, que es el responsable de controlar el flujo de la aplicación, instanciar los beans y servir las vistas activas en cada instancia cliente. Controla los estados del sistema desde que el especialista genera un evento, hasta que el sistema retorna el contenedor correspondiente.

Diagrama de Despliegue.

La figura 7 muestra el diagrama de la configuración del hardware del sistema y los nodos físicos que lo componen. El sistema necesita para su ejecución un Servidor de Aplicaciones, un Servidor de Base de Datos y las terminales web donde cada especialista tendrá acceso al sistema.

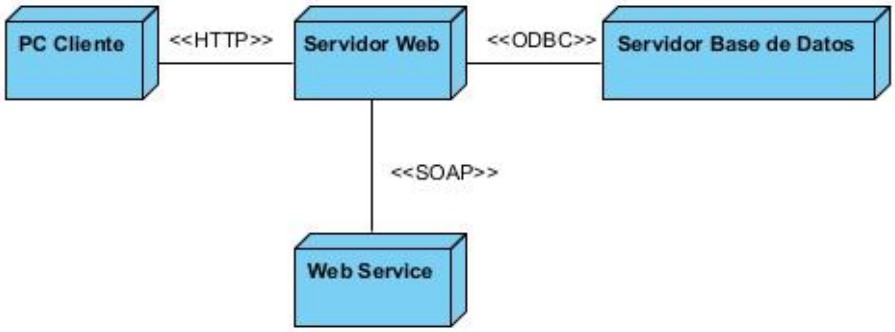


Fig. 6. Diagrama de Despliegue. Se presentan los componentes necesarios para el despliegue del sistema y los protocolos de comunicación entre ellos.

2.6 Diseño de los Casos de Uso del Sistema.

2.6.1 Patrones de Diseño.

Los Patrones de Diseño para la Asignación de Responsabilidades (GRASP, por sus siglas en inglés), son patrones generales de software para asignación de responsabilidades. Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones [32].

Definición de los patrones GRASP.

Patrón	**Descripción**
Bajo acoplamiento	Es la idea de tener las clases lo menos ligadas entre sí que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y

	disminuyendo la dependencia entre las clases. Para determinar el nivel de acoplamiento de clases, son muy buenos los diagramas de colaboración de UML.
Alta cohesión	Nos dice que la información que almacena una clase debe de ser coherente y debe estar relacionada con la clase. Cada elemento de nuestro diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable.
Experto	Este patrón indica que la responsabilidad de hacer cualquier función debe recaer sobre la clase que tenga toda la información necesaria para hacerlo. De este modo se mantiene la información encapsulada y se obtiene un diseño con mayor cohesión.
Creador	El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos. El propósito fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento. Al aplicarlo se da soporte al Bajo Acoplamiento.

Controlador	Es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es el controlador quien recibe los datos del usuario y quien los envía a las distintas clases según el método llamado.

2.6.2 Diagrama de Interacción.

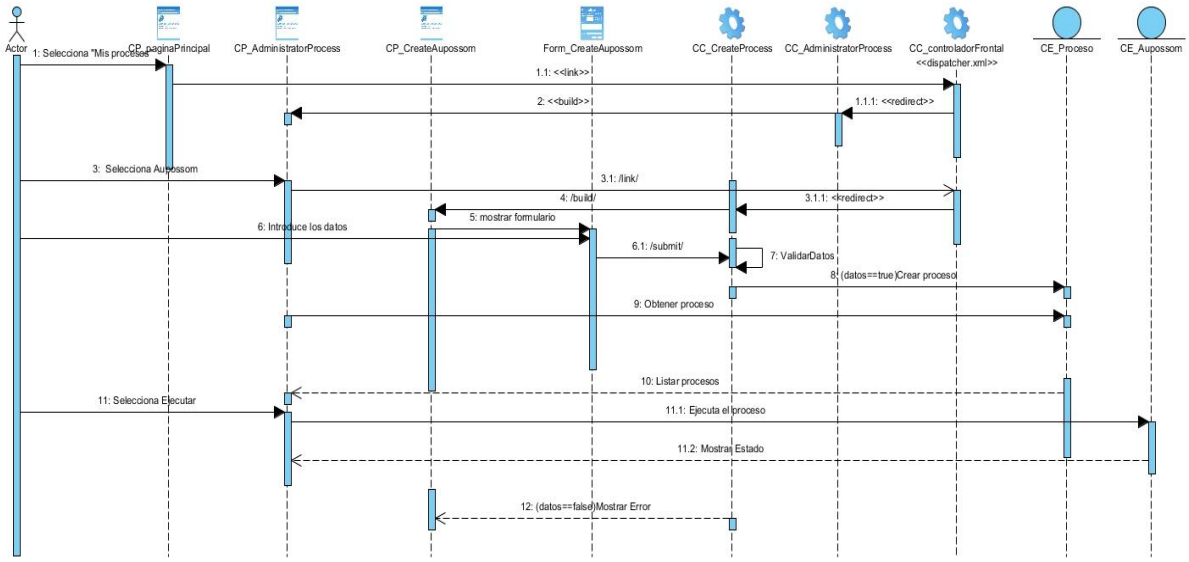


Fig. 7. Diagrama de Secuencia para el CU Realizar Análisis Automático de Poses Derivadas del Docking. Las páginas cliente (CP) se encargan de realizar peticiones a las clases controladoras (CC), estas consultan a las clases entidades (CE) y realizan las llamadas al Servicio Web.
El resto de los Diagramas de Secuencia se encuentran en el anexo del trabajo.

2.6.3 Diagrama de Clases del Diseño.

Diagrama de Clases del Diseño del CU Realizar Análisis Automático de Poses Derivadas del Docking.

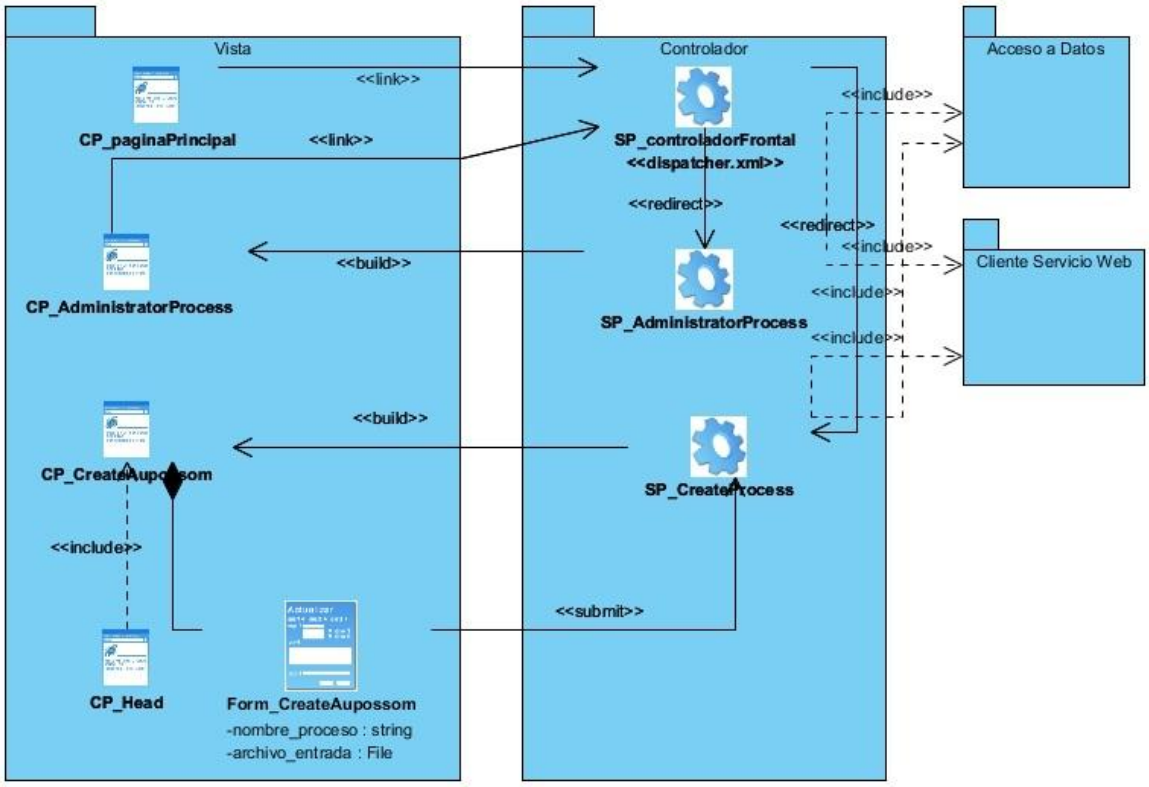


Fig. 8. Diagrama de Clases del Diseño para el CU Realizar Análisis Automático de Poses Derivadas del Docking. Se muestran las páginas clientes (CP) en el paquete Vista, las páginas servidoras (SP) en el paquete Controlador y los paquetes encargados de gestionar los Servicios Web y la Base de Datos.

El resto de los Diagramas de Clases del Diseño se encuentran en el anexo del trabajo.

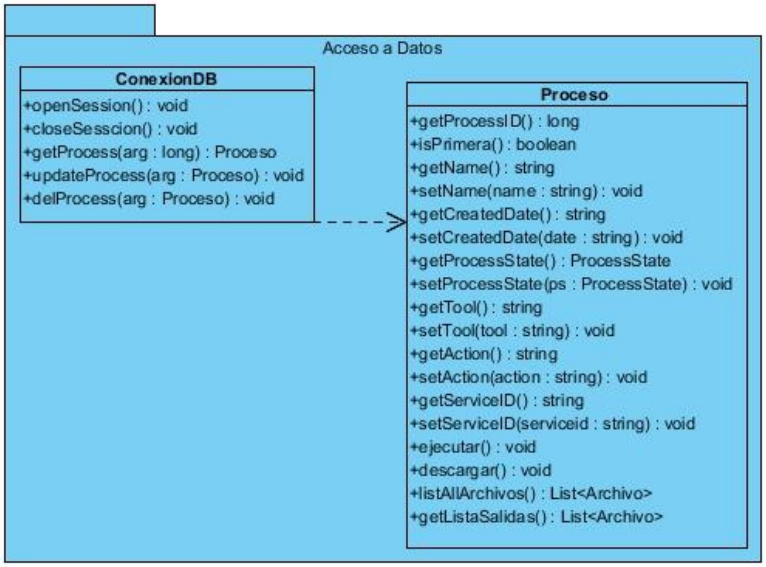


Fig. 9. Diagrama de Clases del Diseño para el paquete de Acceso a Datos. Muestra los componentes encargados de gestionar la base de datos.

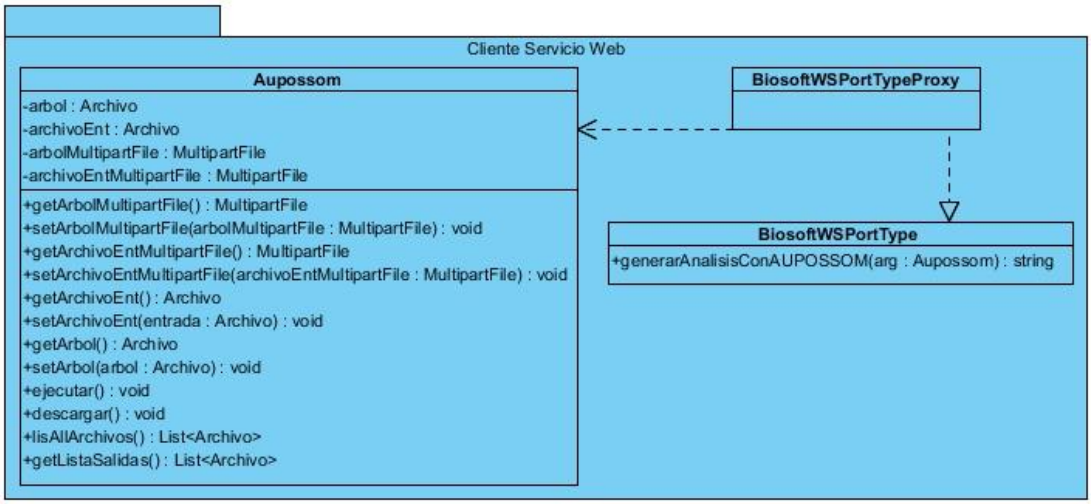


Fig. 10. Diagrama de Clases del Diseño para el paquete Cliente Servicio Web. Este paquete contiene las clases necesarias para invocar los Servicios Web que ejecutan las operaciones del sistema.

2.7 Estructura de la Base de Datos.

Para el desarrollo del Servicio de Análisis Automático de Poses Derivadas del Docking se hace necesaria la creación de una base de datos para el almacenamiento de los hechos de entrada y de salida del servicio. Las tablas contienen instancias de cada uno de los módulos del sistema por lo que contienen un identificador propio, lo que permite un correcto funcionamiento del sistema.

Con la implementación del Caso de Uso Realizar Análisis Automático de Poses Derivadas del Docking se incorpora la tabla: aupossom; en esta nueva tabla se insertan los datos referentes a la corrida que vaya a crear el especialista en ese momento.

Descripción de la tabla Aupossom.

Nombre: aupossom		
Descripción: Registra la información referente a los datos para crear un nuevo proceso de Análisis Automático de Poses Derivadas del Docking.		
Atributo	**Tipo**	**Descripción**

process_id	bigint	Identificador de la tabla.
action	varchar	Indica la acción que realiza ese proceso.
created_date	varchar	Fecha de creación del proceso.
end_date	varchar	Fecha fin del proceso.
name	varchar	Nombre del proceso.
state	varchar	Estado en que se encuentra el proceso.
tool	varchar	Herramienta que se está utilizando en el proceso.
specialist_id	bigint	Identificador del especialista.
archivoent_id_archivo	bigint	Identificador de los archivos de entrada del proceso.
árbol_id_archivo	bigint	Identificador del archivo de salida del proceso.

Las otras tablas creadas son: archivo, que guarda la información necesaria para generar una corrida del especialista así como guardar los archivos resultantes y especialista, que guarda la información necesaria de los usuarios que acceden al sistema.

Las descripciones de los atributos de las otras tablas se encuentran en el anexo del trabajo.

A partir de la relación de cada una de las entidades utilizadas, se genera el siguiente Modelo de Datos.

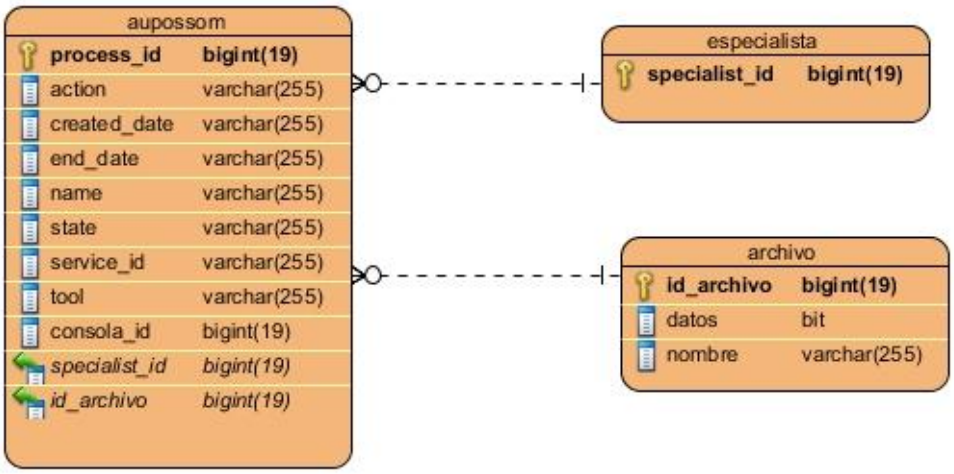


Fig. 11. Modelo de Datos.Contiene las tablas de la Base de Datos que intervienen en el CU Realizar Análisis Automático de Poses Derivadas del Docking.

2.8 Conclusiones del capítulo.

1. Se elaboró el Modelo de Dominio, el cual permitió ofrecer una visión de los objetos que existen o los eventos que suceden en el entorno donde se desarrollará el Servicio de Análisis Automático de Poses Derivadas del Docking.
2. Se definieron los requisitos funcionales y no funcionales, los cuales debe cumplir el sistema fueron y especificados por el cliente.
3. Se modelaron los artefactos correspondientes a la fase de Análisis y Diseño, los cuales constituyen la base de la implementación del producto.
4. Se realizó el Modelo de Datos, el cual posibilitó obtener un mayor conocimiento de las relaciones existentes entre las clases persistentes pertenecientes a la Plataforma de Servicios Bioinformáticos de la UCI para el desarrollo del Servicio de Análisis Automático de Poses Derivadas del Docking.

Capítulo 3: Implementación y Prueba.

En este capítulo se convierten los elementos del diseño en elementos de implementación. Describe los elementos del modelo de diseño y se implementa un servicio web y una interfaz cliente en términos de componentes. También presenta las pantallas de la aplicación como los prototipos de interfaz de usuario, el mapa de navegación así como los casos de prueba realizados sobre la aplicación.

3.1 Diagrama de componentes.

Un Diagrama de Componentes muestra las dependencias lógicas entre componentes software, sean éstos fuentes, binarios o ejecutables, ilustran las piezas del software, controladores, etc. Los Diagramas de Componentes prevalecen en el campo de la arquitectura de software pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema. A continuación se representa el Diagrama de Componentes del Caso de Uso Realizar Análisis Automático de Poses Derivadas del Docking.

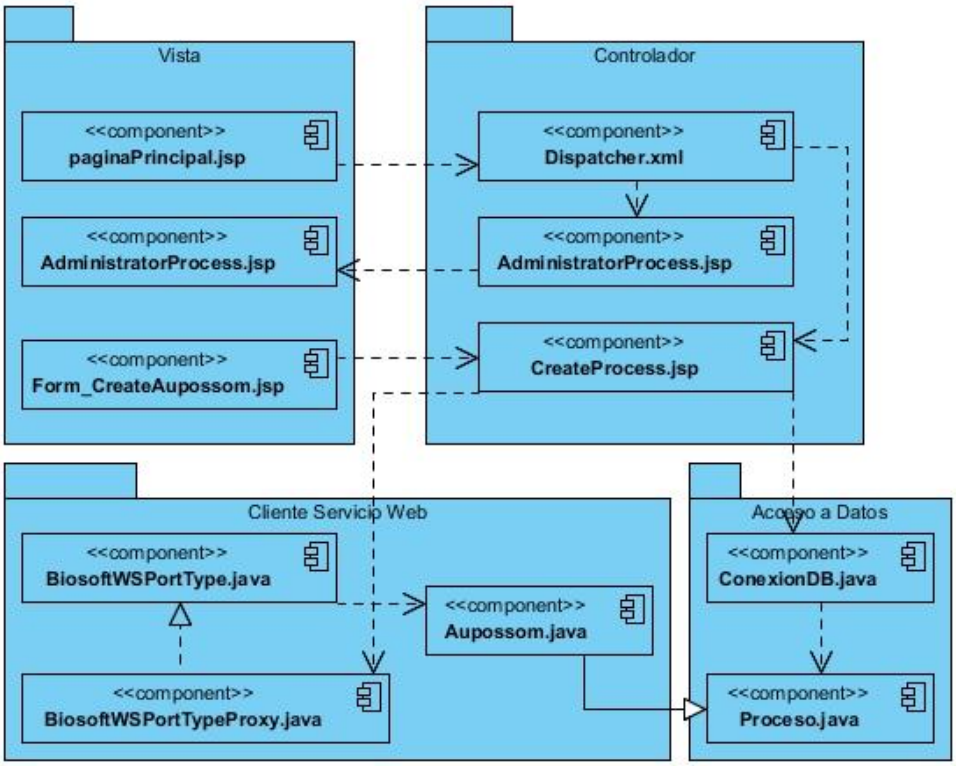


Fig. 12. Diagrama de Componentes del CU Realizar Análisis Automático de Poses Derivadas del Docking. Muestra los componentes del Caso de Uso y las relaciones entre ellos.
El resto de los Diagramas de Componentes se encuentran en el anexo del trabajo.

3.2 Pantallas de la aplicación.

Mapa de navegación.

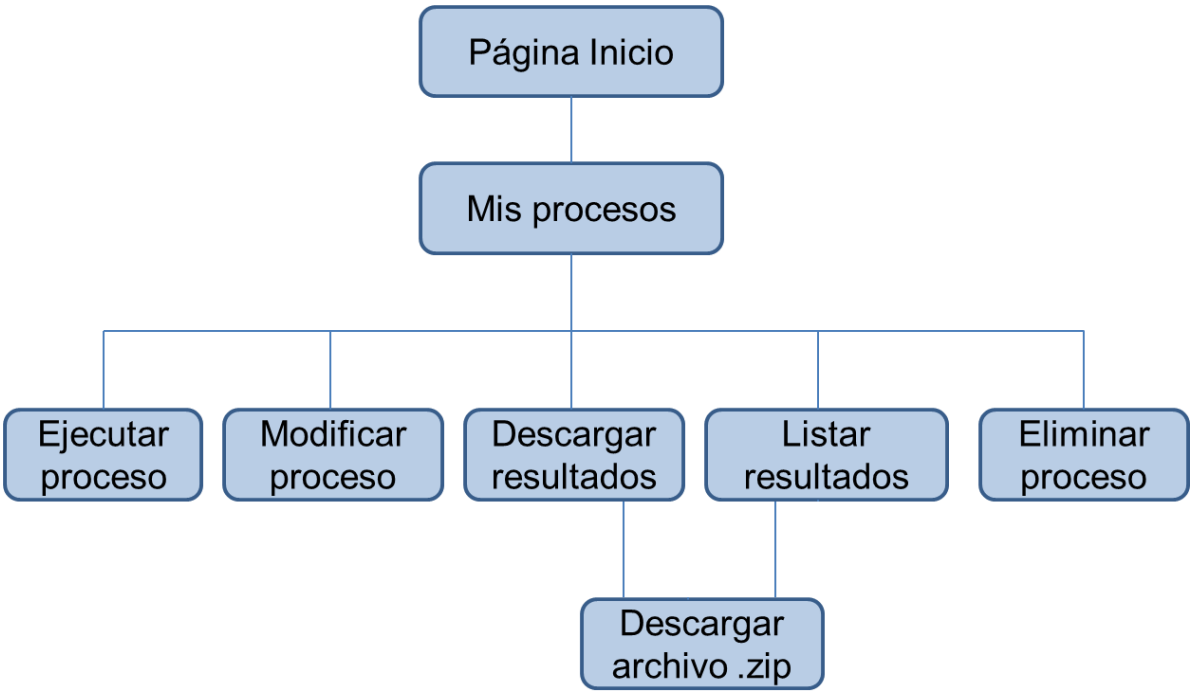


Fig. 13. Mapa de navegación del sistema.

Interfaz de usuario.

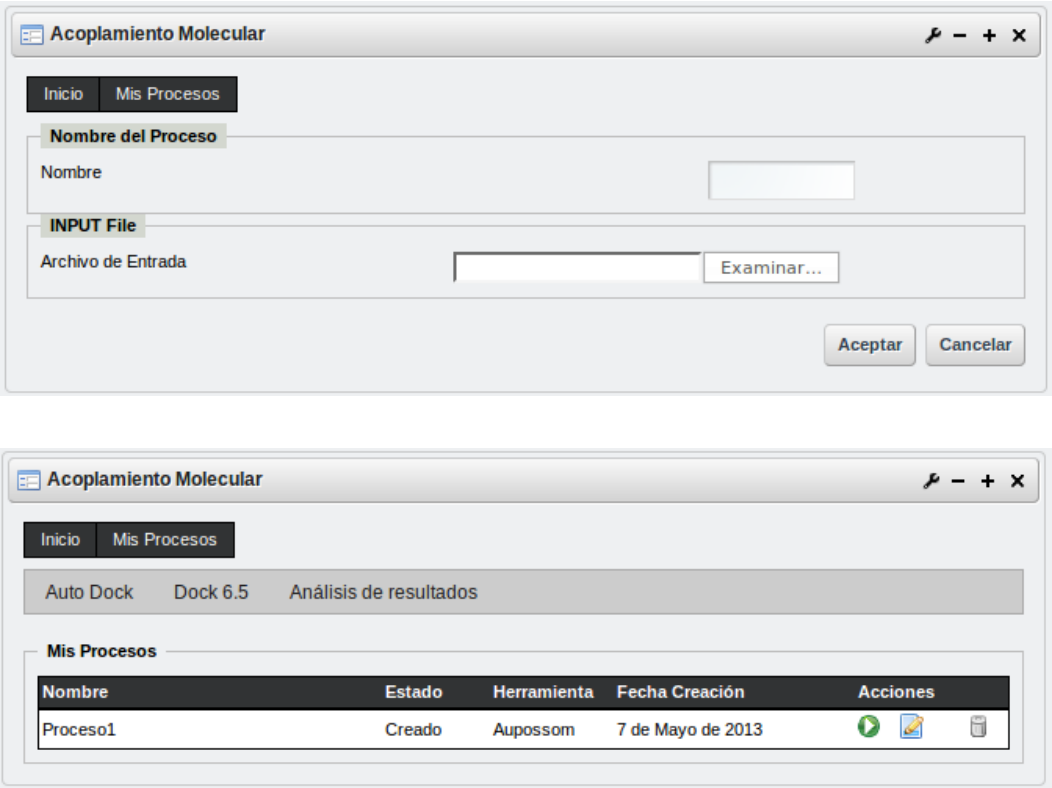


Fig. 14. Interfaz de usuario Realizar Análisis Automático de Poses Derivadas del Docking.

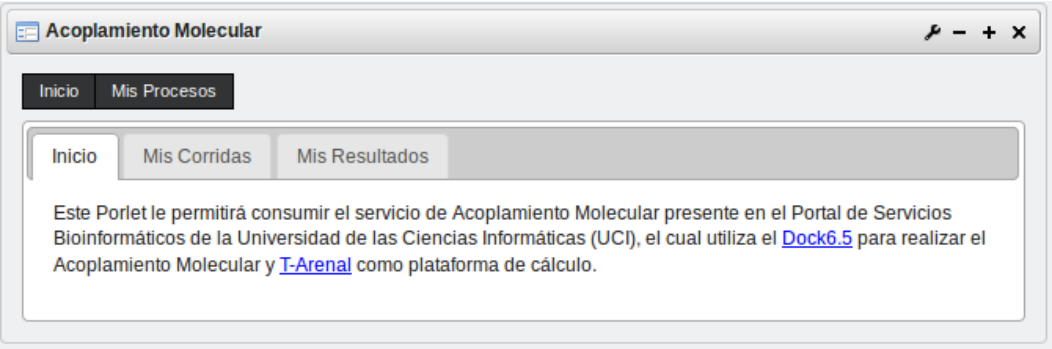


Fig. 15. Interfaz de usuario de la Página Principal.

El resto de las interfaces se encuentran en el anexo del trabajo.

Resultado.

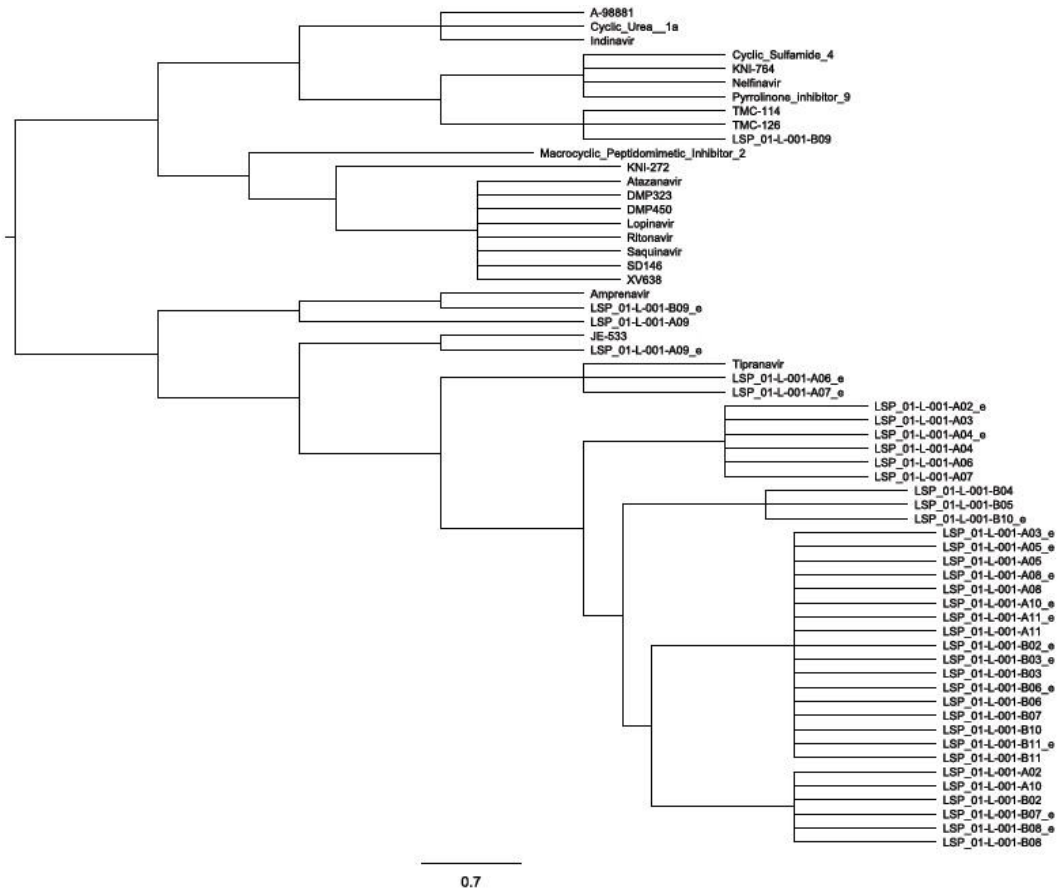


Fig. 16. Archivo de salida del Servicio de Análisis Automático de Poses Derivadas del Docking, representado en forma de árbol y visualizado con el FigTree. Las hojas son los ligandos conocidos y la longitud de las ramas está relacionada con la diferencia entre la huella de las moléculas.

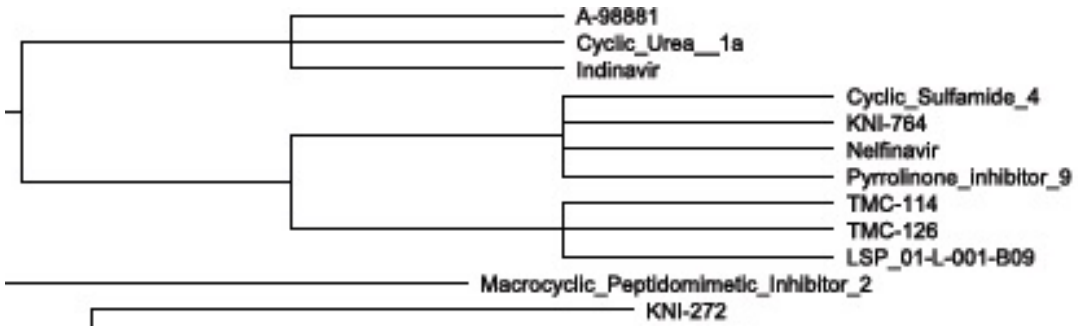


Fig. 17. Fragmento de la Figura 13. El AuPosSOM clusteriza por criterios estructurales y agrupa los ligandos conocidos de acuerdo a diferentes parámetros.

3.3 Casos de prueba.

Las pruebas de caja negra se centran en lo que se espera de un módulo, es decir, intentan encontrar casos en que el módulo no se atiene a su especificación. Mediante la utilización de este método se demuestra que las funciones del software son operativas, la entrada se acepta de forma adecuada, se produce una salida correcta y la integridad de la información externa se mantiene. El nivel de prueba utilizado fue el Desarrollador y la técnica funcional. A continuación se presentan los casos de prueba para el CU Realizar Análisis Automático de Poses Derivadas del Docking.

Descripción de las variables.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Nombre del Proceso	Campo de texto	No	Campo de texto con longitud de 1 a 255 caracteres, sin espacios y no acepta caracteres especiales.
2	Archivo de Entrada	Campo de archivo	No	Campo para la subida del archivo de entrada, solo puede ser .zip.

Sección 1: Caso de prueba Realizar Análisis Automático de Poses Derivadas del Docking.

Escenario	Descripción	Nombre del proceso	Archivo de entrada	Respuesta del sistema	Flujo Central

EC 1.1 Realizar Análisis Automático de Poses Derivadas del Docking utilizando datos correctos.	Se llenan los campos correctamente	V Proceso1	V /home/yas/files/example.zip	El sistema valida los datos introducidos y si son correctos crea el nuevo proceso en la base de datos.	1. El usuario accede al sistema. 2. Introduce el nombre del proceso, sube el archivo de entrada y presiona el botón Aceptar. 3. El sistema crea el nuevo proceso en la base de datos. 4. El usuario presiona el botón Ejecutar del nuevo proceso. 5. El sistema analiza las poses y genera un archivo en forma de árbol.
EC 1.2 Realizar Análisis Automático de Poses Derivadas del Docking utilizando datos vacíos.	Se introducen los campo en blanco	V Proceso1	NA	El sistema valida los datos introducidos y si existen campos en blanco, el sistema muestra un mensaje mostrando el error en la interfaz.	1. El usuario accede al sistema. 2. Introduce el nombre del proceso y presiona el botón Aceptar. 3. El sistema reconoce que no se han subido archivos de entrada y muestra un mensaje mostrando el error en la interfaz.

EC 1.3 Realizar Análisis Automático de Poses Derivadas del Docking utilizando datos incorrectos.	Se introducen campos inválidos	I Proceso1 &%$	V /home/yas/files/example.zip	El sistema valida los datos introducidos y si existen campos incorrectos, el sistema muestra un mensaje mostrando el error en la interfaz.	1. El usuario accede al sistema. 2. Introduce el nombre del proceso, sube el archivo de entrada y presiona el botón Aceptar. 3. El sistema reconoce que el campo “Nombre del Proceso” contiene caracteres inválidos y muestra un mensaje mostrando el error en la interfaz.
EC 1.4 Realizar Análisis Automático de Poses Derivadas del Docking utilizando datos incorrectos.	Se introducen campos inválidos	I Proceso1	V /home/yas/files/example.pdb	El sistema valida los datos introducidos y si existen campos incorrectos, el sistema muestra un mensaje mostrando el error en la interfaz.	1. El usuario accede al sistema. 2. Introduce el nombre del proceso, sube el archivo de entrada y presiona el botón Aceptar. 3. El sistema reconoce que el campo “Archivo de Entrada” tiene una extensión incorrecta y muestra un mensaje mostrando el error en la interfaz.

3.4 Conclusiones del capítulo.

1. Se implementó el sistema en términos de componentes y se realizó el Diagrama de Componentes, logrando el resultado del diseño.
2. Se presentaron las interfaces de usuario de la aplicación y se elaboró el mapa de navegación del sistema.
3. Se realizaron y ejecutaron cuatro casos de prueba para validar el Servicio Web de Análisis Automático de Poses Derivadas del Docking y el Portlet Cliente, obteniéndose los resultados esperados.

Conclusiones.

1. Se implementó un Servicio Web para realizar el Análisis Automático de Poses Derivadas del Docking.
2. Se agregaron las clases y funcionalidades necesarias al Portlet Acoplamiento Molecular para consumir el Servicio Web permitiéndole al especialista Administrar sus procesos y Visualizar, Listar y Descargar los resultados dados por el sistema.
3. Se realizó el análisis y el diseño del servicio de Análisis Automático de Poses Derivadas del Docking y del Portlet Cliente, obteniéndose los artefactos correspondientes a cada fase de la metodología propuesta.
4. Se agregaron las tablas necesarias a la Base de Datos del Portlet Acoplamiento Molecular, para el almacenamiento de los datos que se generan durante la ejecución del sistema.

Recomendaciones.

Incorporar una herramienta al Módulo de Acoplamiento Molecular de la Plataforma de Servicios Bioinformáticos para convertir los archivos de salida del servicio de Docking a formato PDB para el posterior análisis con el AuPosSOM.

Referencias bibliográficas.

[1] Díaz León, A. Módulo básico de la Plataforma de Servicios Bioinformáticos. Tesis de Diploma. Universidad de las Ciencias Informáticas. La Habana. 2012.

[2] Martínez Pérez, O.; Díaz León, A.; Agramonte Delgado, A. UCIBioSoft: Portal web de servicios bioinformáticos. VIII Congreso Internacional de Informática en la salud. 201.

[3] Padilla, J. & Rojo, A. Simulación del reconocimiento entre proteínas y moléculas orgánicas o docking. Aplicación al diseño de fármacos. 2002.

[4] Scior, T.; Martínez E. & Salinas E. Los modelos in silico. Una herramienta para el conocimiento farmacológico. 2007.

[5] Trott, O. & Olson, A. J. AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. Journal of Computationalo Chemistry. 2009. 31 (2): p. 455-461.

[6] The Official UCSF DOCK Web-site. Disponible en: http://dock.compbio.ucsf.edu/DOCK_6/index.html.

[7] Tsui, B. PRIME 2011 Osaka University. 2011.

[8] Bouvier, G. & Bertho, G. AuPosSOM : Automatic analysis of Poses using Self-Organizing Map. 2009.

[9] Mantsyzov, A.; Bouvier, G.; Evrard-Todeschi & Bertho, G. AuPosSOM: New Approach for the Identification of Active Compounds in the Set of Docked Molecules. 2011. Disponible en: http://auposom.org.

[10] The Apache Software Foundation. Apache Tomcat - Welcome! [en línea]. 2012 [Consultado en: diciembre 2012]. Disponible en: http://tomcat.apache.org/index.html.

[11] de León Estupiñan, J. Portales y Portlets Web. 2011. Disponible en: http://www.slideshare.net/jossydeleon/portales_y_porlets_web_9377907.

[12] Liferay Inc. Portal, Content, and Collaboration for the Enterprise [en línea].2012 [Consultado en: diciembre 2012]. Disponible en: http://www.liferay.com/products/liferay_portal/overview.

[13] Johnson, R & Hoeller, J. The Spring Framework. 2008. 590 p.

[14] Go-Pivotal Inc. SPRING FRAMEWORK [En línea]. 2012 [Consultado en: diciembre 2012]. Disponible en: http://www.springsource.org/spring-framework.

[15] Perera, S. Axis2, Middleware for Next Generation Web Services. En: International Conference on Web Services ICWS '06. 2006. p. 833-840.

[16] King, G; Bauer, C; Rydhal Andersen, M; Enmanuel, B & Ebersole, S. HIBERNATE - Persistencia relacional para Java idiomático [En línea]. 2004 [Consultado en: diciembre 2012]. Disponible en: http://docs.jboss.org/hibernate/core/3.5/reference/es-ES/html_single/.

[17] Arnold, K; Gosling, J & Holmes, D. El lenguaje de Programación Java. 2005. 354 p.

[18] Blancos, C. Entornos de Desarrollo Integrado [En línea]. 2012 [Consultado en: diciembre 2012]. Disponible en: http://carlosblanco.pro/2012/04/entornos-desarrollo-integrado-introduccion/.

[19] Geer, D. Eclipse becomes the dominant Java IDE. En: Computer. 2005. P. 16-18.

[20] Balduino, R. Introduction to OpenUP [En línea]. 2007 [Consultado en: diciembre 2012]. Disponible en: http://www.eclipse.org/epf/general/OpenUP.pdf.

[21] Popkin Software and Systems. Modelado de Sistemas con UML [En línea]. [Consultado en: diciembre 2012]. Disponible en: http://es.tldp.org/Tutoriales/doc-modelado-sistemas-UML/doc-modelado-sistemas-uml.pdf.

[22] Universidad Politécnica de Valencia. Introducción a Herramientas CASE y System Architect [En línea]. [Consultado en: diciembre 2012]. Disponible en: http://users.dsic.upv.es/asignaturas/eui/mtp/doc-practicas/intro_case_SA.pdf.

[23] Targetware Informática S.A.C. Visual Paradigm para UML [En línea]. 2012 [Consultado en: diciembre 2012]. Disponible en: http://www.software.com.ar/visual-paradigm-para-uml.html.

[24] UML CASE tool for software development. Disponible en: http://www.visual-paradigm.com/product/vpuml/.

[25] Sánchez, J. Sistemas Gestores de Bases de Datos. 2009. 182 p.

[26] Ramakrishnan, R & Gehrke, J. Database management system [En línea]. 2007 [Consultado en: diciembre 2012]. Disponible en: http://pages.cs.wisc.edu/~dbbook/.

[27] The PostgreSQL Global Development Group. PostgreSQL 8.1 Press Kit [En línea]. 2011 [Consultado en: diciembre 2012]. Disponible en: http://www.postgresql.org/about/press/presskit91/es/.

[28] Larman, C. Modelo del Dominio. 2003. 23 p.

[29] Weitzenfeld, A. Ingeniería de software orientada a objetos con UML. 2005. 678 p.

[30] Tello Cáceres, J. Diagrama de Casos de Uso [En línea]. [Consultado en: diciembre 2012]. Disponible en: http://www2.uah.es/jcaceres/capsulas/DiagramaCasosDeUso.pdf.

[31] Márquez Gómez, J. Arquitectura MVC Visión General [En línea]. 2011 [Consultado en: diciembre 2012] .Disponible en: http://jorge.queideas.com/wp-content/uploads/2011/11/Arquitectura-MVC.pdf.

[32] Visconti, M. & Astudillo, H. Fundamentos de Ingeniería de Software [En línea]. 2008 [Consultado en: diciembre 2012]. Disponible en: http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf.

Anexos

A. Descripciones de los CU del Sistema.

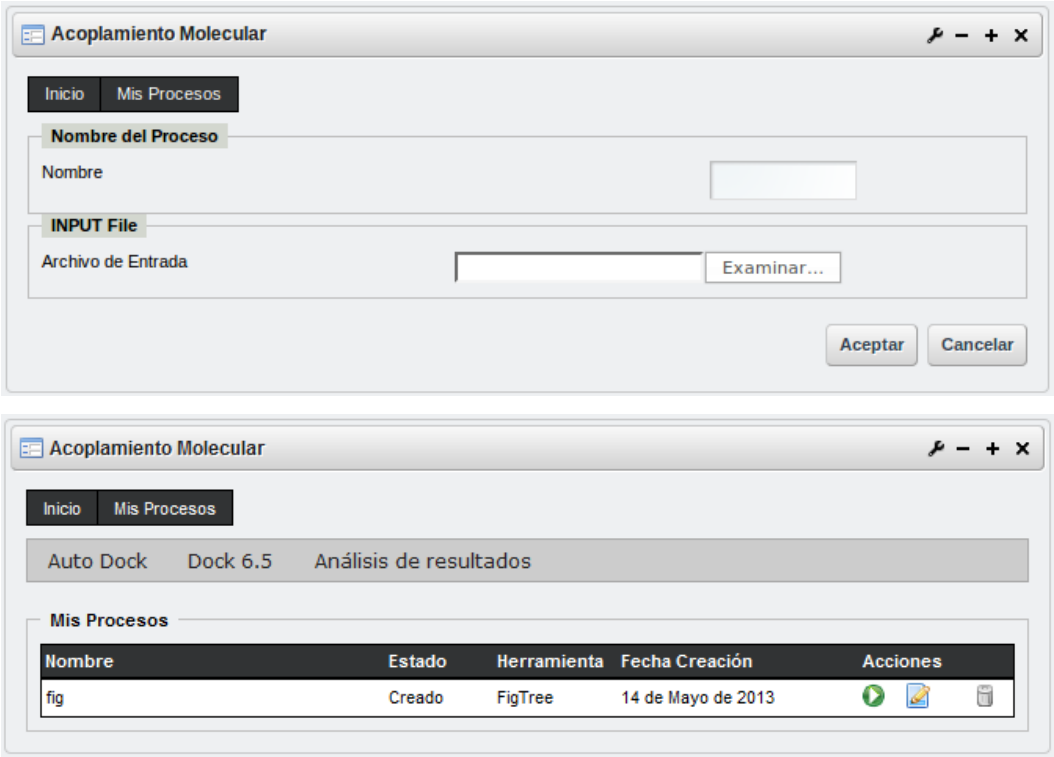
Descripción del CU Visualizar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.

Caso de Uso:	Visualizar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.
Actores:	Especialista
Resumen:	El caso de uso comienza cuando el especialista necesita visualizar los ficheros de salida del Servicio de Análisis Automático de Poses Derivadas del Docking. El sistema brinda un formulario en el cual se sube el archivo de entrada para visualizar el árbol en formato NEWICK. El caso de uso finaliza cuando se genera el archivo de salida en formato PDF.
Precondiciones:	
Referencias	RF6.
Prioridad	Crítica.

Flujo Normal de Eventos		
Sección "Visualizar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking"		
Acción del Actor		**Respuesta del Sistema**
1.	El especialista inicia el caso de uso cuando selecciona la opción "Análisis de resultados"-->"Visualizar resultados" en la interfaz principal.	
2.		Muestra en la interfaz un formulario con los siguientes campos: • Nombre del proceso. • Archivos de entrada.
3.	Introduce los datos necesarios (nombre del proceso y archivos de entrada) para la creación del proceso y presiona el botón "Aceptar".	
4.		El sistema valida los datos entrados y si son

		correctos crea el nuevo proceso en la base de datos.
5.	El usuario presiona “Ejecutar” en el nuevo proceso creado.	
6.		El sistema convierte el archivo TREE a un formato de imagen en forma de árbol y genera un fichero de salida en formato PDF.
7.		Fin del proceso.
Flujos Alternos		
		3.1. Si los datos entrados son incorrectos, el sistema muestra un mensaje mostrando el error en la interfaz.

Prototipo de Interfaz



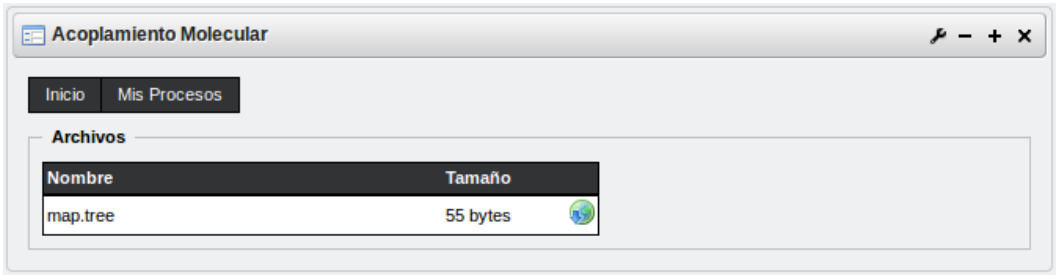
Descripción del CU Listar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.

Caso de Uso:	Listar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.
Actores:	Especialista

Resumen:	El caso de uso comienza cuando el especialista necesita listar los ficheros de salida del proceso de Análisis Automático de Poses Derivadas del Docking. El sistema brinda una interfaz en la cual se listan todos los ficheros de salida de un proceso de un especialista.
Precondiciones:	El estado del proceso debe ser “Terminado”.
Referencias	RF5.
Prioridad	Secundaria.

Flujo Normal de Eventos		
Sección “Listar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking”		
Acción del Actor		**Respuesta del Sistema**
1.	El especialista inicia el caso de uso cuando presiona el botón “Mis procesos” en la interfaz principal.	
2.		Muestra en la interfaz una tabla con los procesos de un especialista.
3.	El especialista presiona el botón “Listar” en el proceso que desea observar los ficheros de salida.	
4.		El sistema muestra una interfaz con todos los archivos de salida de ese proceso.
Flujos Alternos		

Prototipo de Interfaz



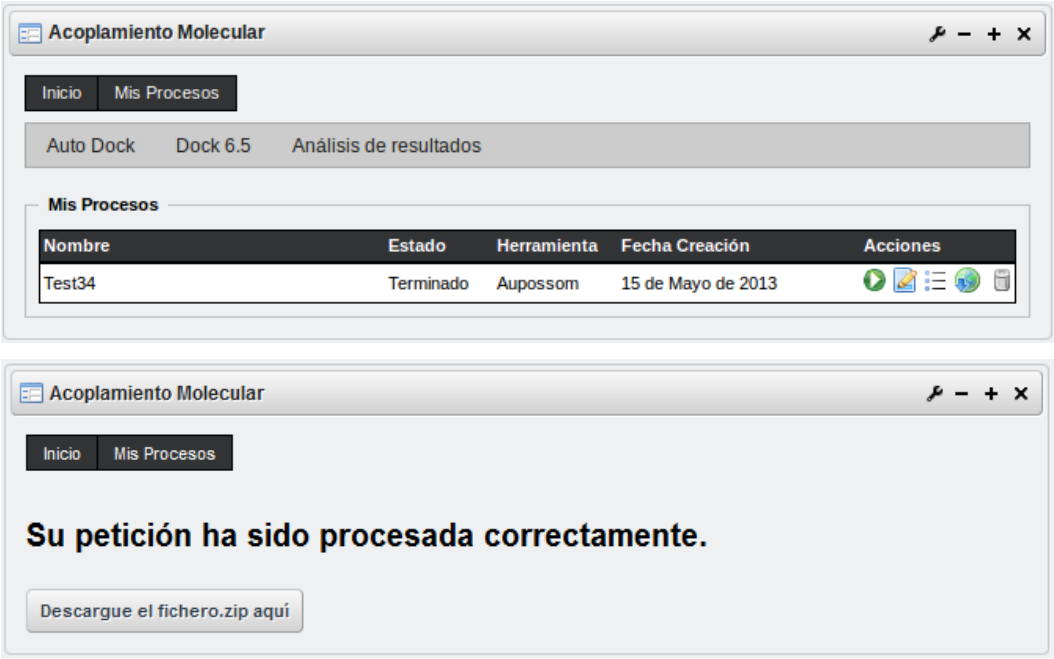
Descripción del CU Descargar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.

Caso de Uso:	Descargar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.
Actores:	Especialista
Resumen:	El caso de uso comienza cuando el especialista necesita descargar los ficheros de salida del Servicio de Análisis Automático de Poses Derivadas del Docking. El sistema brinda un formulario en el cual se descargan los ficheros de salida de un proceso en formato de archivo comprimido ZIP. El caso de uso finaliza cuando se descarga el archivo ZIP.
Precondiciones:	El estado del proceso debe ser “Terminado”.
Referencias	RF7.
Prioridad	Secundaria.

Flujo Normal de Eventos		
Sección “Descargar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking”		
Acción del Actor		**Respuesta del Sistema**
1.	El especialista inicia el caso de uso cuando selecciona la opción “Mis Procesos” en la interfaz principal.	
2.		Muestra en la interfaz una tabla con todos los procesos de un especialista.
3.	El usuario presiona el botón “Descargar” en el proceso que desea almacenar en su PC los archivos de salida.	

4.		El sistema muestra un formulario con el botón “Descargue el fichero .zip aquí”.
5.	El usuario presiona el botón “Descargue el fichero .zip aquí”.	
6.		El sistema descarga los archivos de salida comprimidos en formato ZIP en la PC del especialista.
7.		Fin del proceso.
Flujos Alternos		

Prototipo de Interfaz



Descripción del CU Administrar proceso de Análisis Automático de Poses Derivadas del Docking.

Caso de Uso:	Administrar proceso de Análisis Automático de Poses Derivadas del Docking.
Actores:	Especialista
Resumen:	El caso de uso comienza cuando el especialista accede a la página principal del módulo Acoplamiento Molecular, donde puede elegir opciones como Modificar, Listar y Eliminar los procesos de Análisis Automático de Poses Derivadas del Docking. El caso de uso finaliza

	cuando se obtiene el resultado de la opción deseada.	
Precondiciones:	El especialista debe estar autenticado y tener al menos un proceso en la base de datos.	
Referencias	RF2, RF3, RF4.	
Prioridad	Secundaria.	
Flujo Normal de Eventos		
1.	El especialista inicia el caso de uso cuando selecciona la opción "Mis Procesos" en la interfaz principal.	
2.		El sistema muestra en la interfaz una tabla con todos los procesos de un especialista. Cada proceso con las siguientes opciones: • Modificar proceso. • Eliminar proceso.
3.	El usuario escoge una opción.	
4.		Si escoge: • Modificar proceso, ver sección Modificar proceso de Análisis Automático de Poses Derivadas del Docking. • Eliminar proceso, ver sección Eliminar proceso de Análisis Automático de Poses Derivadas del Docking.
Sección "Listar proceso de Análisis Automático de Poses Derivadas del Docking"		
Acción del Actor		**Respuesta del Sistema**
1.	El especialista selecciona la opción "Mis Procesos" en la interfaz principal.	
2.		El sistema muestra en la interfaz una tabla con todos los procesos de un especialista.
Sección "Modificar proceso de Análisis Automático de Poses Derivadas del Docking"		
Acción del Actor		**Respuesta del Sistema**
1.	El especialista presiona el botón "Modificar proceso" en el proceso que desea alterar.	

2.		El sistema Muestra en la interfaz un formulario con los siguientes campos: • Nombre del proceso. • Archivos de entrada.
3.	Introduce los datos necesarios (nombre del proceso y archivos de entrada) para la creación del proceso y presiona el botón “Aceptar”.	
4.		El sistema valida los datos entrados y si son correctos modifica el proceso seleccionado.
3.		
Flujo Alterno de la Sección “Modificar proceso de Análisis Automático de Poses Derivadas del Docking”		
		3.1. Si los datos entrados son incorrectos, el sistema muestra un mensaje mostrando el error en la interfaz.
Sección “Eliminar proceso de Análisis Automático de Poses Derivadas del Docking”		
Acción del Actor		**Respuesta del Sistema**
1.	El especialista presiona el botón “Eliminar proceso” en el proceso que desea descartar.	
2.		El sistema elimina el proceso de la base de datos.

Prototipo de Interfaz

B. Diagramas de Secuencia.

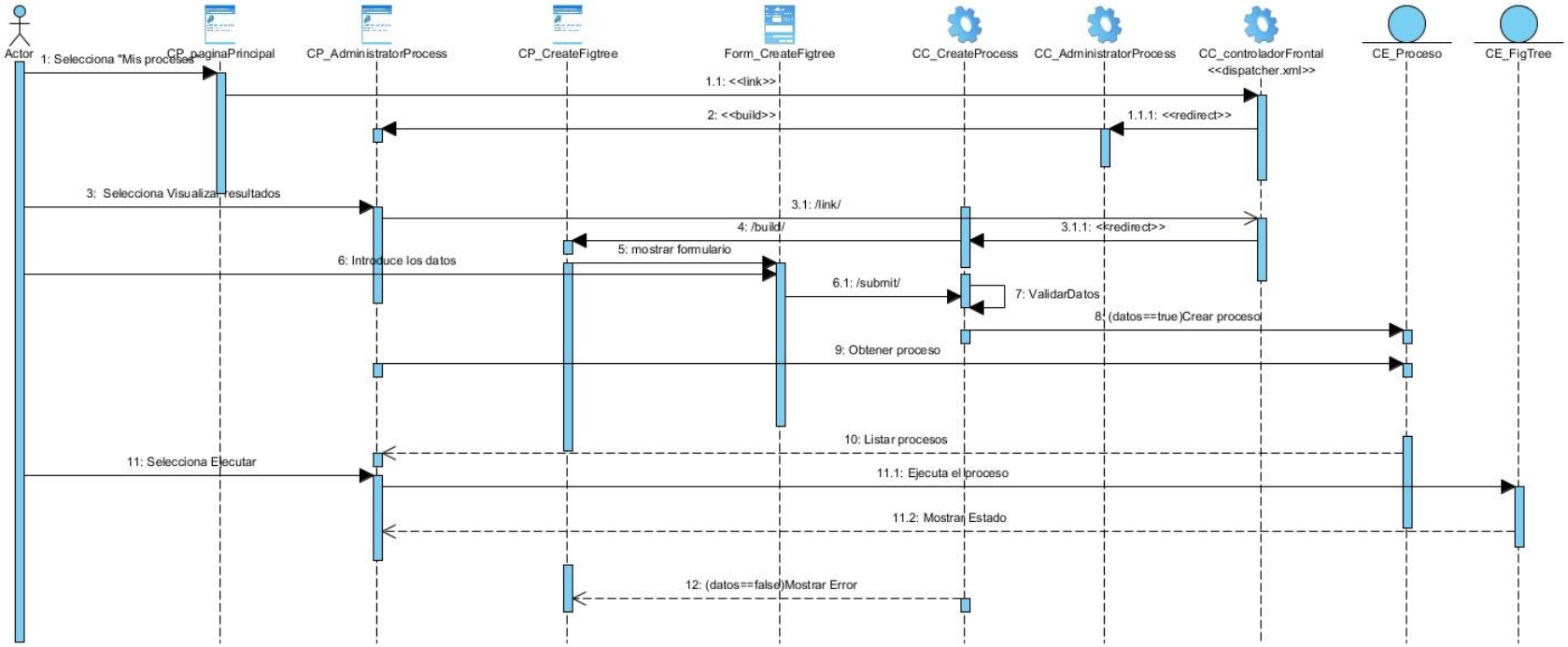


Fig. 18. Diagrama de Secuencia para el CU Visualizar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.

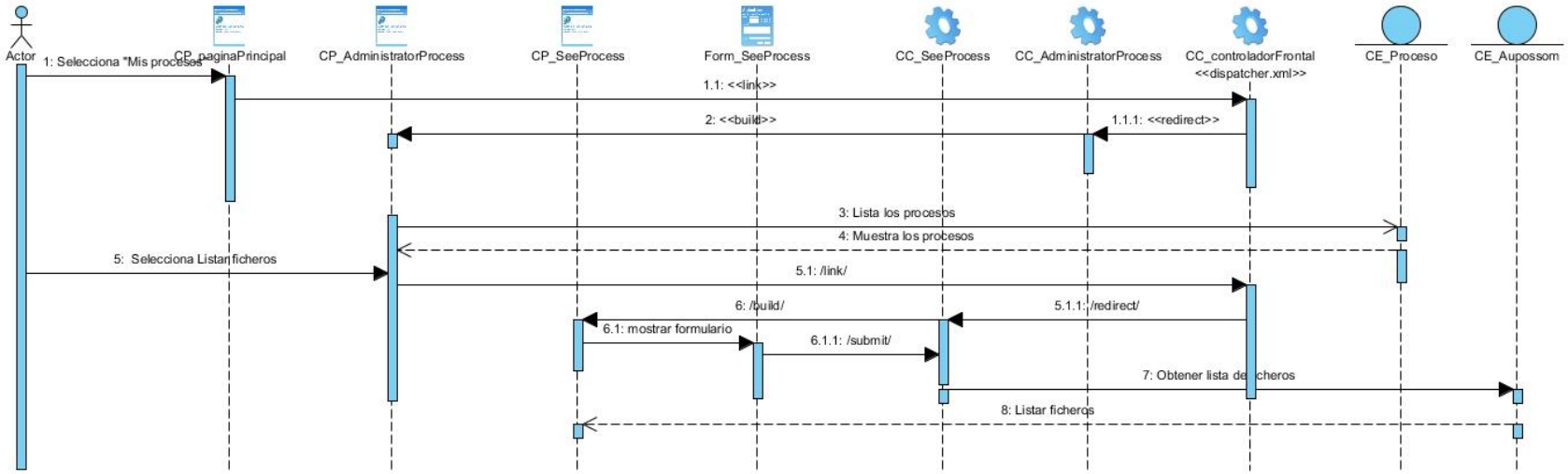


Fig. 19. Diagrama de Secuencia para el CU Listar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.

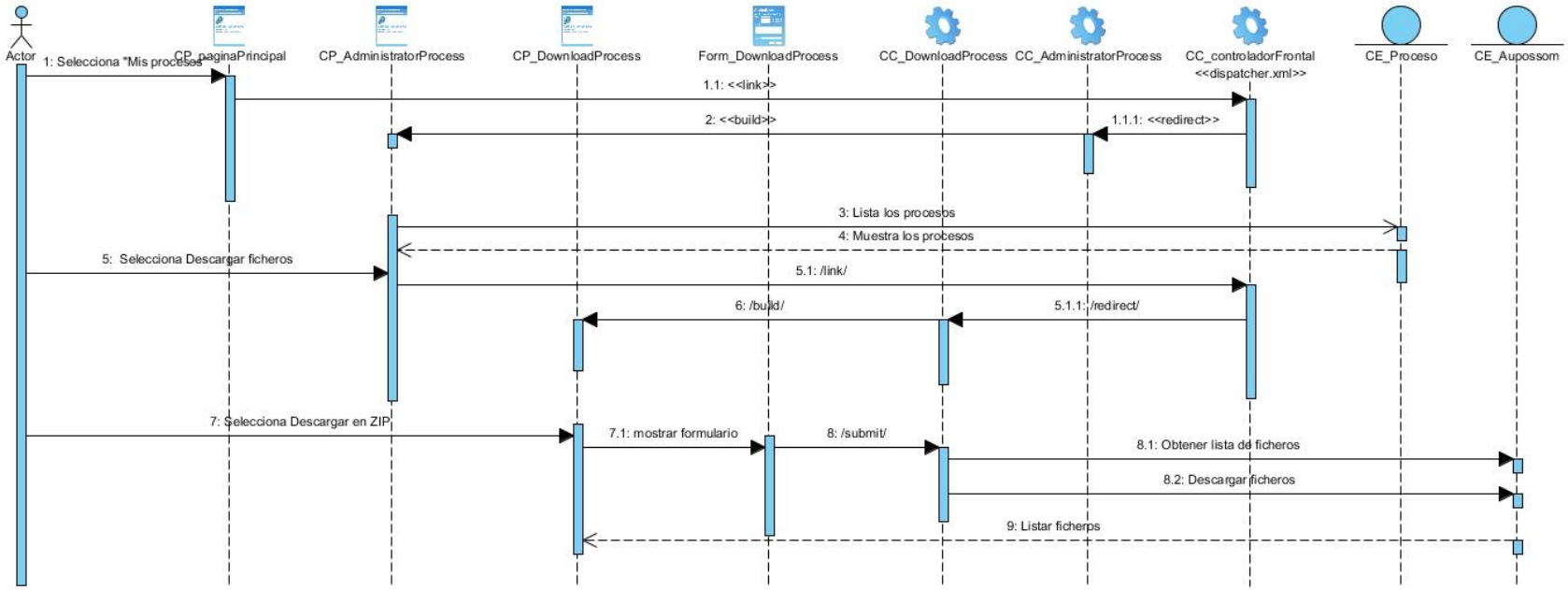


Fig. 20. Diagrama de Secuencia para el CU Descargar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.

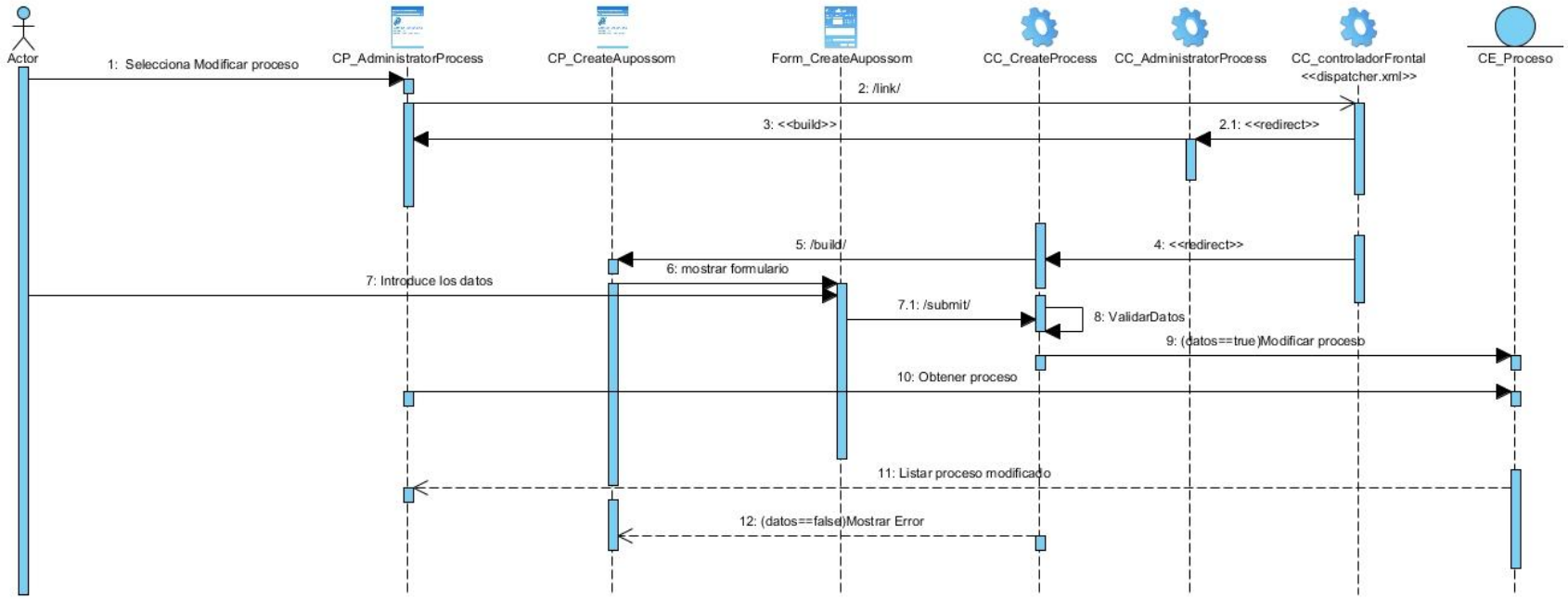


Fig. 21. Diagrama de Secuencia del Escenario Modificar proceso de Análisis Automático de Poses Derivadas del Docking para el CU Administar proceso de Análisis Automático de Poses Derivadas del Docking.

C. Diagramas de Clases del Diseño.

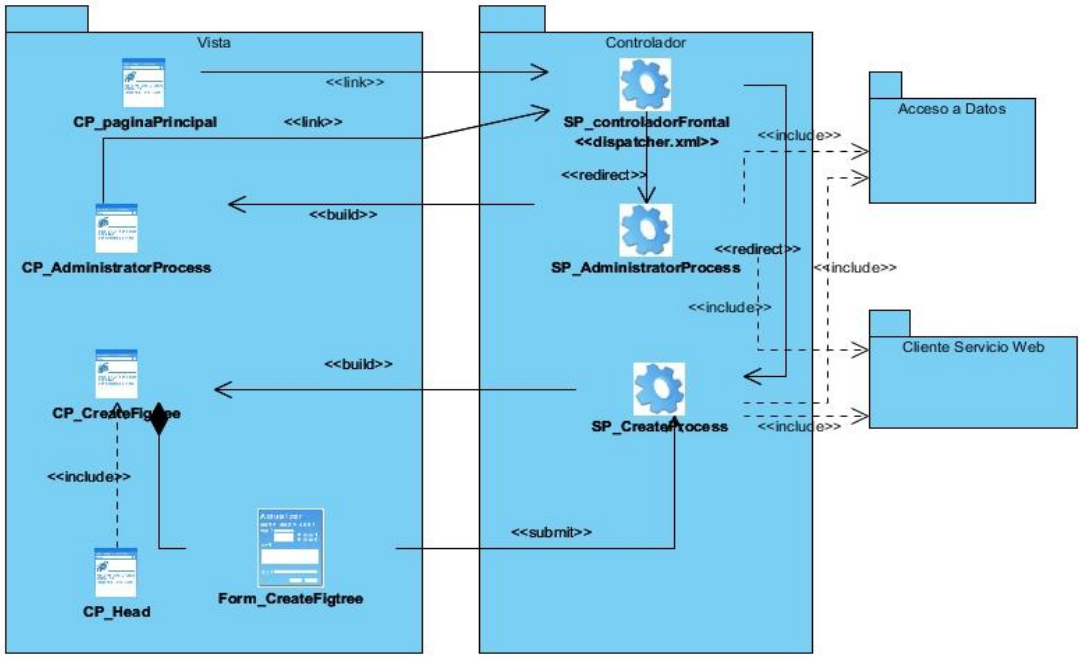


Fig. 22. Diagrama de Clases del Diseño para el CU Visualizar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.

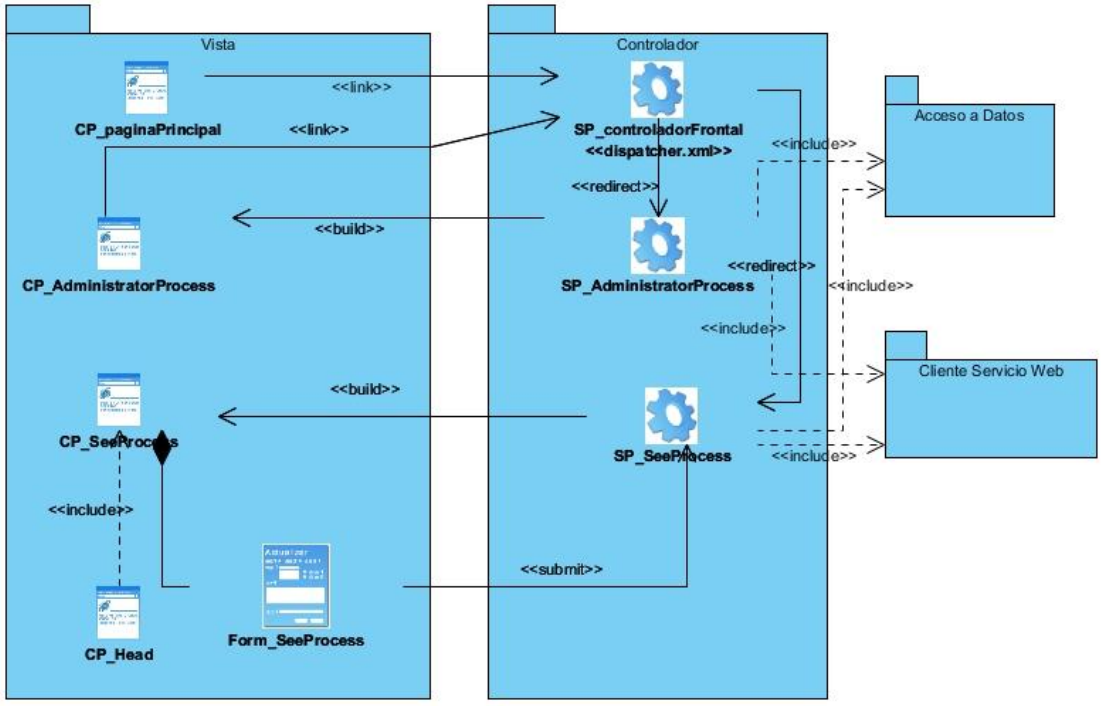


Fig. 22. Diagrama de Clases del Diseño para el CU Listar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.

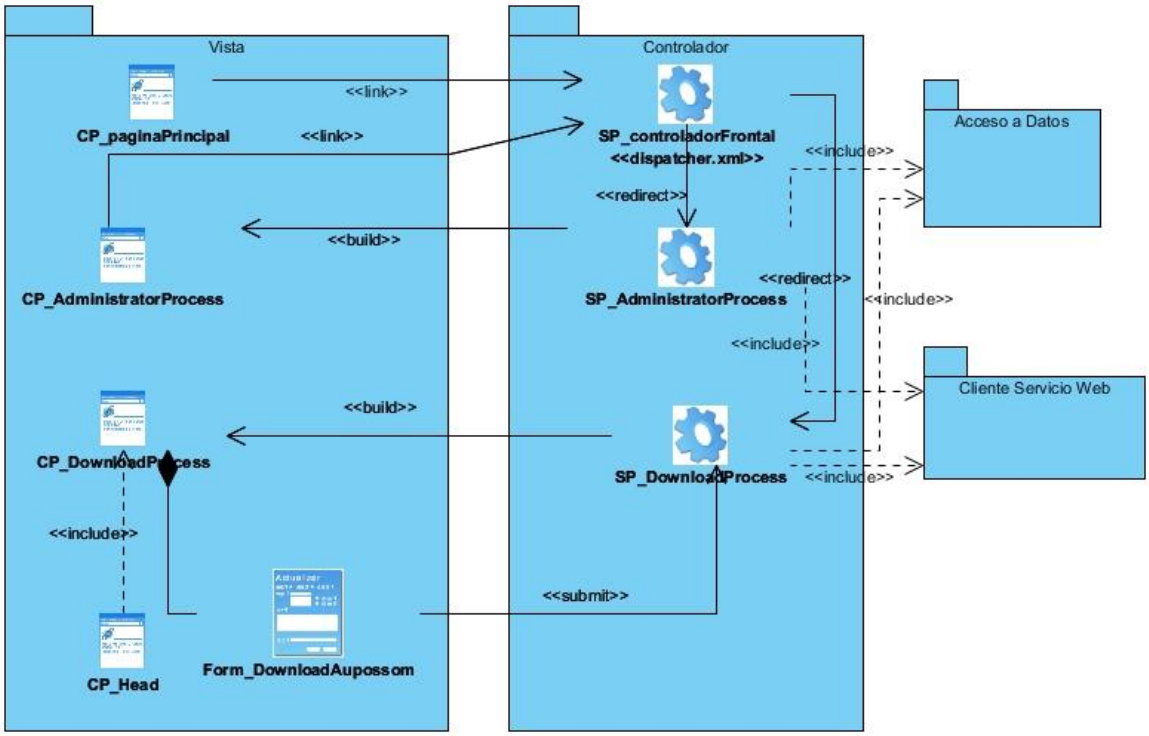


Fig. 23. Diagrama de Clases del Diseño para el CU Descargar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.

D. Diagramas de Componentes.

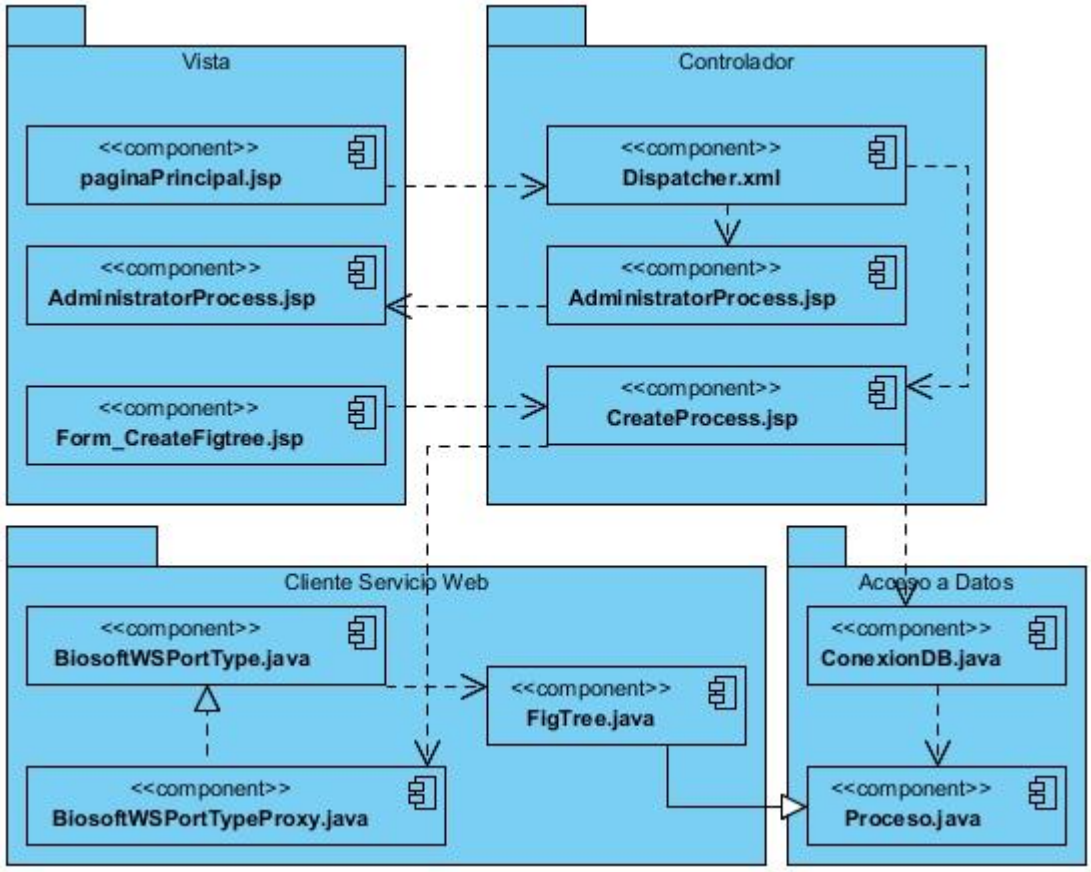


Fig. 24. Diagrama de Componentes para el CU Visualizar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.

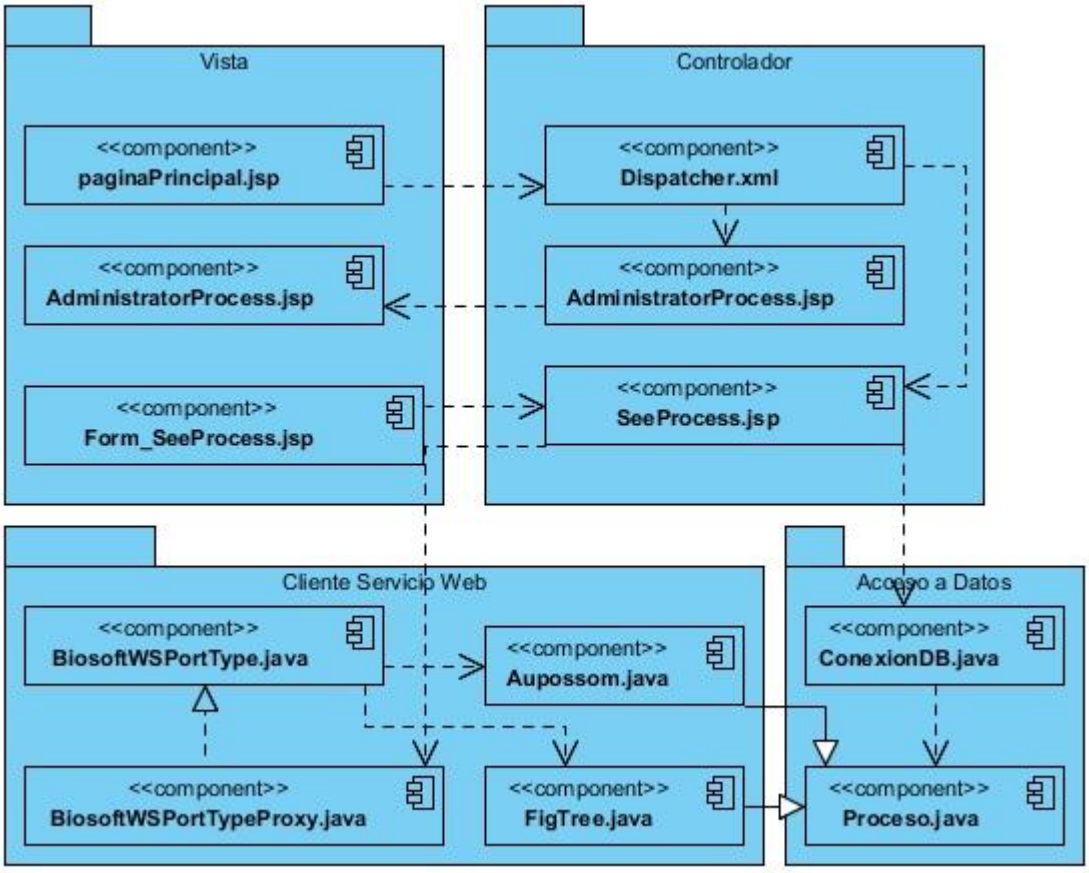


Fig. 25. Diagrama de Componentes para el CU Listar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.

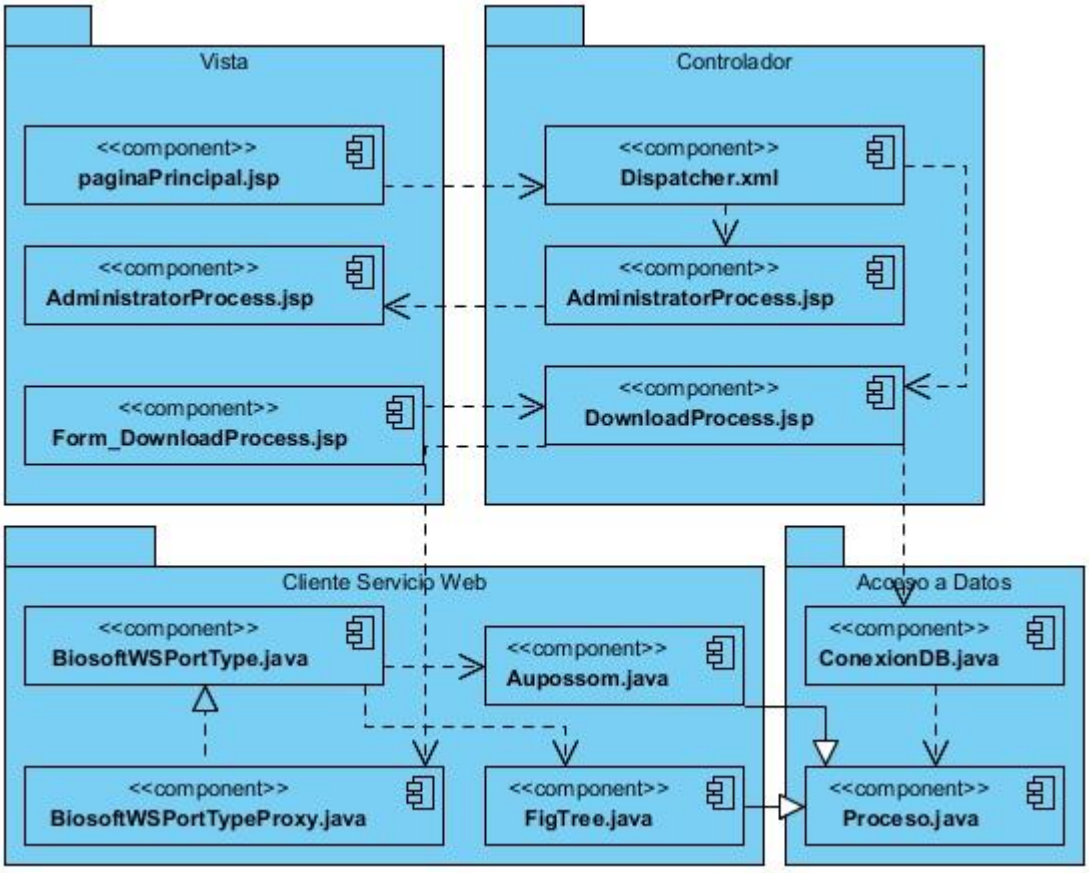


Fig. 26. Diagrama de Componentes para el CU Descargar ficheros del proceso de Análisis Automático de Poses Derivadas del Docking.

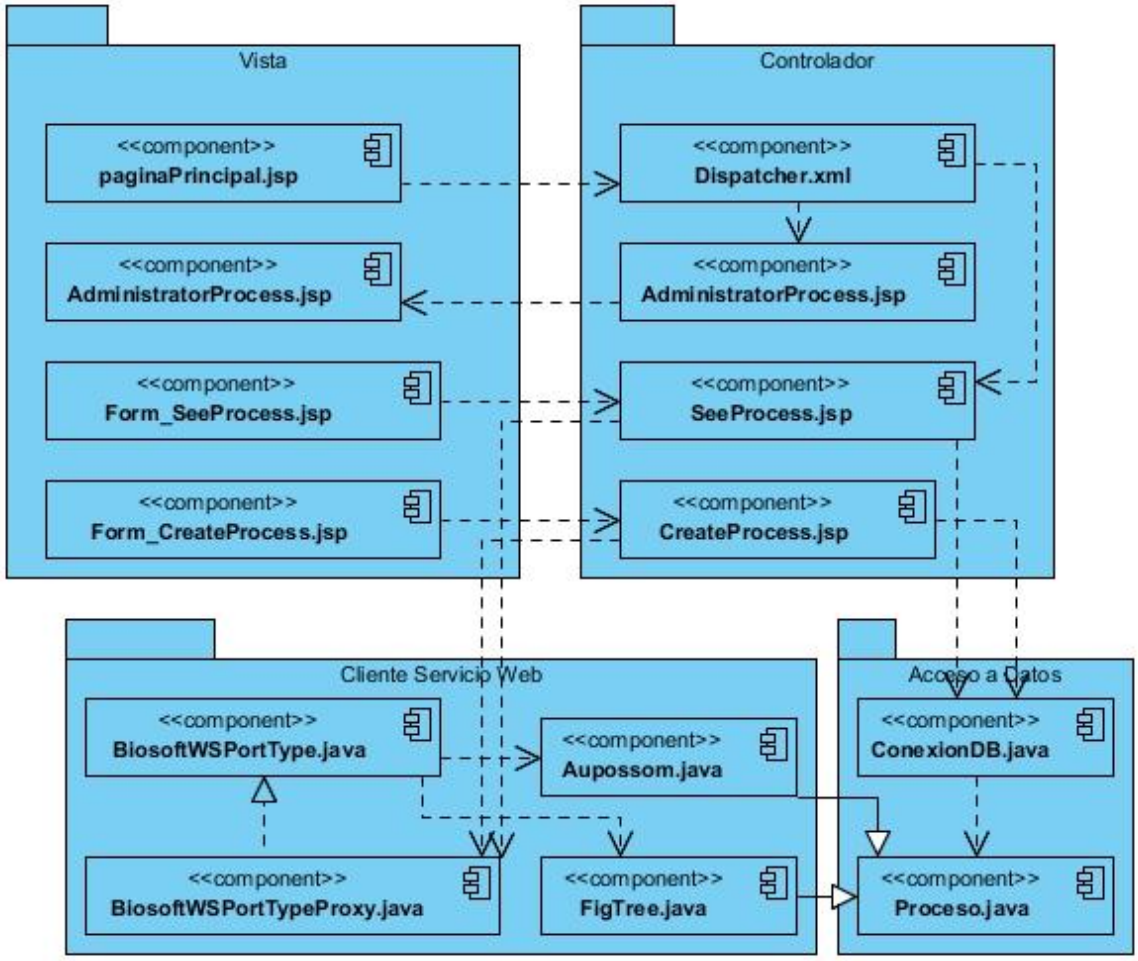


Fig. 27. Diagrama de Componentes para el CU Administrar proceso de Análisis Automático de Poses Derivadas del Docking.