Universidad de las Ciencias Informáticas

Facultad 6

Título: "Portal Web de Servicios Bioinformaticos"

Trabajo de Diploma para optar por el título de Ingeniero Informático

Autor(es): Niurka Martínez Durán

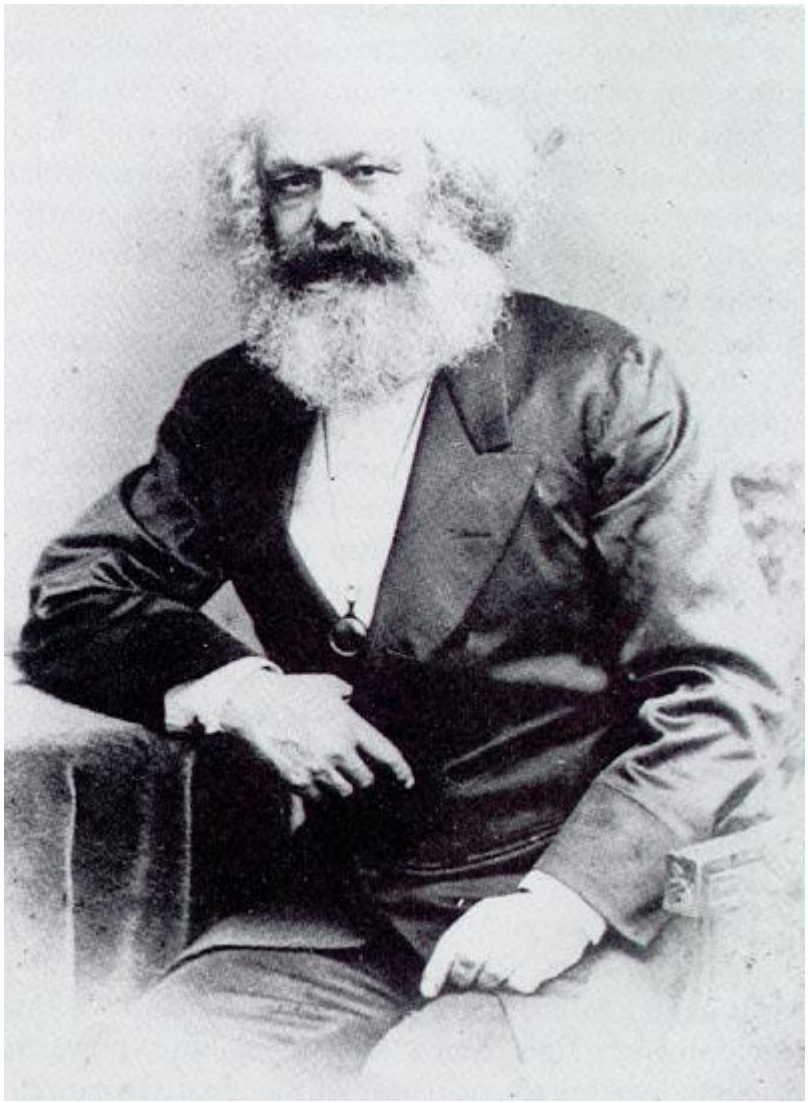
Yaumara Mercedes Hernández Álvarez

Tutor(es): MSc. Noel Moreno Lemus

MSc.Longendri Aguilera Mendoza

La Habana, junio de 2008

"Año del 50 Aniversario de la Revolución"



“En la ciencia no existen calzadas reales, y quien aspire a remontar sus luminosas cumbres; ha de estar dispuesto a escalar la montaña por senderos escabrosos”.

Karl Marx

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 se firma la presente a los ____ días del mes de ____ del año _____.

Yaumara M. Hernández Álvarez

_____

Firma del Autor

MSc. Noel Moreno Lemus

_____

Firma del Tutor

Niurka Martínez Duran

_____

Firma del Autor

MSc. Longendri Aguilera Mendoza

_____

Firma del Tutor

Tutores

Msc. Noel Moreno Lemus

Universidad de las Ciencias Informáticas, La Habana, Cuba.

Email: noel@uci.cu

Msc. Longendri Aguilera Mendoza

Universidad de las Ciencias Informáticas, La Habana, Cuba.

Email: loge@uci.cu

A la Revolución, a Fidel y a la UCI, por dejarnos formar parte de este proyecto futuro.
A nuestros tutores, por preocuparse tanto por nuestra tesis y ayudarnos cada vez que lo necesitábamos.
A nuestros amigos especialmente a Keiler González, Edel Moreno Lemus, Jorge Luis Cuellar y Daniel Mirayes.

Yaumara

….. En especial a mi mamá.
…..A mis tres hermanos, Yaimara, Jose Luis y Leyanit.
…..A mi tía María Rosa.
…..A mi abuela, que es la persona mas maravillosa que conozco.
…..A Fina que es como mi abuelita.
…..A Anileidi Verdecia la esposa de mi hermano.
…..A el papá de mi hermano Jose Luis que ha sido maravilloso conmigo.
….. También a mis tía Elba Rosa e Isolina Rosa.

En fin, a todos lo que hicieron posible este sueño realidad, por haber confiado en mí, por demostrar que todo llega algún día.

Niurka

…..A mi papa Carlos Manuel Martínez Jiménez. Por haberme ayudado llegar hasta aquí.
…..A mi abuela Claribel y mi tía Maribel.

Con infinito amor…
Dedico esta tesis en primer lugar a mi abuelo Rafael Ángel que aunque ya no este presente se que su sueño era que yo me realizara como profesional.

A mis padres, mis hermanos, especialmente mi hermana Yai y Jose Luis, a mi abuelita, y mis tías, principalmente mi tía María Rosa, al padre de mi hermano y a Anileidi su esposa.

Yaumara

Con todo el amor del mundo…
Le dedico mi tesis en especial a mi padre Carlos Manuel Martínez Jiménez. por ser tan especial y único conmigo.
A mis abuelas Claribel y Gladis.
A mis tías en especial a Maribel y Nurdis.
A mi mamita y mis hermanos Carlitin y Greysy.
A mis amigas que estuvieron junto conmigo: Dariela, Ade, Male y Yadi.

Niurka

El Grupo de Bioinformática de la UCI necesita agrupar de una forma centralizada todos los proyectos realizado por el mismo. En la actualidad no se cuenta con un Portal Web al que puedan acceder los investigadores en busca del software desarrollado por el grupo, por tal motivo se requiere de un Portal Web que agrupe las aplicaciones de servicios desarrolladas. Teniendo en cuenta esta necesidad, el propósito fundamental de esta investigación es desarrollar un Portal Web que permita dar solución a este problema. El sistema fue llevado a cabo siguiendo los pasos que propone el Proceso Unificado de Desarrollo de Software y empleándose para el modelado del mismo la herramienta CASE Visual Paradigm, se utilizaron tecnologías de software libre: java como plataforma y lenguaje de programación.

Palabra Clave: Portal Web.

Introducción

Desde su surgimiento, la era computacional ha ayudado a la ciencia en sus diferentes ramas. Actualmente muchos fenómenos hasta entonces desconocidos son descubiertos, investigados y conducidos a resultados en ordenadores por todo el mundo. Varios de ellos se deben a la apasionante disciplina Bioinformática, una impresionante fusión entre la Biología y la Informática.

El Centro Nacional para la Información Biotecnológica "National Center for Biotechnology Information" (NCBI por sus siglas en Inglés, 2001) define: [1]

"Bioinformática es un campo de la ciencia en el cual confluyen varias disciplinas tales como: biología, computación y tecnología de la información. El fin último de este campo es facilitar el descubrimiento de nuevas ideas biológicas así como crear perspectivas globales a partir de las cuales se puedan discernir principios unificadores en biología. (...)
(...)El proceso de analizar e interpretar los datos es conocido como biocomputación. Dentro de la Bioinformática y la biocomputación existen otras sub-disciplinas importantes:

- El desarrollo e implementación de herramientas que permitan el acceso, uso y manejo de varios tipos de información
- El desarrollo de nuevos algoritmos (fórmulas matemáticas) y estadísticos con los cuales se pueda relacionar partes de un conjunto enorme de datos, como por ejemplo métodos para localizar un gen dentro de una secuencia, predecir estructura o función de proteínas y poder agrupar secuencias de proteínas en familias relacionadas."

La brillante capacidad de cálculo y la manipulación de grandes cantidades de datos que ofrece la informática han facilitado numerosos avances en el estudio del genoma humano y otras especies menos complejas como las bacterias, levaduras y ratones. También ha posibilitado la detección y el tratamiento de numerosas enfermedades y la elaboración amplia de nuevos fármacos. Otros puntos de desarrollo son el alineamiento de secuencias, la predicción de estructuras proteicas y las redes metabólicas.

De manera general podemos decir que esta disciplina científica emergente se nutre eficazmente de otras tales como la ingeniería genética, la industria bioquímica y farmacéutica, medicina, etc., todas ellas se integran formando todo un equipo multidisciplinario en post del mejoramiento de la vida humana.

Cuba, desde la década de los años 80 inició el fomento y desarrollo de la biotecnología. Hoy en día la Bioinformática tiene un nuestro país un vertiginoso avance y una organización estructurada, existen instituciones como Centro Nacional de Bioinformática (BIOINFO) del CITMA, el Centro de Ingeniería Genética y Biotecnología (CIGB), además de un grupo de coordinación que lo integra entre muchos otros el Polo Científico del Oeste de la capital.

Según el Dr. Juan Pedro Febles (**febles@bioinfo.cu**), director de BIOINFO, el plan estratégico para el desarrollo de la Bioinformática en Cuba estuvo sustentado, fundamentalmente, en:

1. La formación de recursos humanos.

2. Creación de centros de Bioinformática.

3. Desarrollo de clústeres (agrupamiento) de computadoras

4. Realización de congresos y eventos sobre la especialidad.

La Universidad de las Ciencias Informáticas (UCI), con el objetivo de contribuir al avance de la Bioinformática en Cuba ha creado un grupo el cual vincula sus recursos de cómputo disponibles con el conocimiento de estudiantes, profesores y otros profesionales.

El grupo de Bioinformática de la UCI, ha elaborado un conjunto de proyectos en la búsqueda de respuestas a investigaciones relacionadas con este campo tales como:

- Bionova.
- BioGrid.
- BioSis.
- GRAph-TOol

El acceso a estos proyectos se realiza de forma independiente por lo que se convierten en un problema para las investigaciones científicas porque no se puede acceder a los servicios que estos prestan simultáneamente sino de forma independiente.

A partir de un análisis de como utilizar tecnología de la información para organizar, y distribuir estas aplicaciones surge el siguiente **problema**:
¿Cómo agrupar las aplicaciones de servicios desarrolladas por el Grupo de Bioinformática de la UCI?
Para ello el problema se enmarca en el **objeto de estudio**: Aplicaciones de Servicios Bioinformáticos que involucra como **campo de acción:** Aplicaciones Web de Servicios Bioinformáticos.
El **objetivo general** para dar solución a la problemática planteada es: Desarrollar un Portal Web para los servicios del Grupo de Bioinformática de la UCI, del cual se derivan los siguientes **objetivos específicos**:

- Realizar el análisis del Portal Web.
- Diseñar el Portal Web.
- Definir una guía para incorporar nuevos servicios o aplicaciones.
- Desarrollar un Servicio Bioinformático para el Portal Web.

Para cumplir con el objetivo propuesto se llevaron a cabo una serie de **tareas** tales como:

- Análisis del estado del arte de las aplicaciones de Servicios Bioinformáticos.
- Determinación de los requisitos del Portal Web.
- Estudio y selección de las herramientas a utilizar para desarrollar el Portal Web.
- Identificación de los servicios que va a brindar el Portal Web.
- Definición de los estilos gráficos.
- Estructuración del mapa de navegación.
- Estudio y documentación del proceso para incorporar nuevos servicios.
- Implementación de un Servicio Bioinformático.
- Despliegue del Portal Web seleccionado.

De manera general se pretende con este trabajo asegurar un Portal Web de servicios de interfaz amigable y ampliamente definida que satisfaga las necesidades del Grupo de Bioinformática.

Estructuración del contenido.
El presente documento se estructura en cuatro capítulos

Capítulo 1. Fundamentación teórica.
Se realiza el análisis del estado del arte de los Aplicaciones de Servicios Bioinformáticos y se introduce el concepto Servicios Bioinformáticos así como se caracteriza la metodología OpenUP y los distintos software utilizados.

Capítulo 2. Características del sistema.
En el presente capítulo se hace un análisis de la propuesta del sistema, para ello se especifican los requisitos funcionales y no funcionales, se identifica mediante el diagrama de casos de uso del sistema las relaciones entre los actores y casos de uso del sistema; así como las descripciones textuales de cada caso de uso.

Capítulo 3. Análisis y diseño del sistema.

Análisis y Diseño del Sistema. En este capítulo se enfatiza en la descripción de estilos arquitectónicos y patrones de diseño para ello se describen los patrones y se evidencia su utilización .También se tiene en cuenta los diagramas de clases, diagramas de interacción y el modelo de despliegue.

Capítulo 4. Implementación.

Implementación. Este capítulo comienza con el resultado del diseño, implementando el sistema en términos de componentes. Se realiza el diagrama componentes. También se presenta el código fuente de las principales clases y pantallas de la aplicación como por ejemplo prototipos de interfaz de usuario y el mapa de navegación y se incluye una guía para adicionar portlets a la plataforma.

Resultados esperados: Se pretende obtener un Portal Web, que satisfaga las necesidades del Grupo de Bioinformática de la UCI, acorde con los estándares de diseño y presentación de aplicaciones Web.

Capítulo 1 Fundamentación Teórica

Introducción.

En este capítulo se aborda lo referente a los conceptos de Servicios, Aplicaciones Web, Computación Grid y Portlets. Se realiza un análisis completo del estado del arte de las aplicaciones de Servicios Bioinformáticos. Además se brinda información sobre la metodología a emplear, las herramientas y los patrones de diseño y de arquitectura a utilizar.

1.1 Conceptos

1.1.1 Servicios

El continuo avance de la sociedad mundial ha traído paralelamente el incremento de las prestaciones de servicios en todas sus áreas.

Según la Definición establecida en la Serie de normas ISO 9000:

"Un servicio es el resultado de llevar a cabo necesariamente al menos una actividad en la interfaz entre el proveedor y el cliente y generalmente es intangible. La prestación de un servicio puede implicar, por ejemplo: [2]

- una actividad realizada sobre un producto tangible suministrado por el cliente (Terapia genética).
- una actividad realizada sobre un producto intangible suministrado por el cliente (por ejemplo, la entrega de datos reales de análisis en la sangre para una investigación de sus componentes)
- la entrega de un producto intangible (por ejemplo, resultados estadísticos sobre lineamientos generales en la secuencia del ADN).
- la creación de una ambientación para el cliente (por ejemplo, en un laboratorio científico).

En la Informática los servicios han tenido múltiples terminologías, hay quien dice: [3]

"incluyen tanto los servicios profesionales vinculados a la instalación, mantenimiento, desarrollo, integración, etc. de software, como los de soporte técnico de hardware."
Se puede decir entonces que los Servicios Bioinformáticos son aquellos servicios informáticos vinculados directamente con la disciplina científica Bioinformática.

1.1.2 Aplicaciones Web:

El concepto de aplicación Web difiere de cada desarrollador, pero de manera general se podría decir que es un sistema informático que permite al usuario acceder a través de una interfaz interactiva a los recursos y servicios ofrecidos por un servidor web, puede utilizarse como enlace una intranet o Internet. Esta comunicación no se trata solamente de páginas web estáticas, sino que incluye páginas web dinámicas, por tanto, requieren de tecnologías que proporcionen la lógica de la generación de contenido dinámico del lado del servidor (php, asp, jsp). Utilizan además lenguajes interpretados del lado del cliente (HTML, XML, Ajax) y se combinan para aprovechar su potencial y posibilitar una solución más extensible y portable. Pueden ser clasificadas de diferentes formas:

- Informacionales: periódicos, catálogos, manuales, libros electrónicos.
- Interactivas: Formularios de registros, presentación de información personalizada.
- Transaccionales: tienda electrónica, bancos online.
- Workflow: planificación online, monitoreo, gerencia de inventario.
- Comunitarias: Chat, mercados, subastas online.
- Portales: tiendas electrónicas.

1.1.2.1 Portal Web

Un portal es una aplicación web que provee una interfaz común a una comunidad de usuarios determinada para asegurar un único punto de acceso a los servicios y recursos de su interés, entre los que se pueden encontrar, chats, foros, buscadores, compra electrónica, correo electrónico, aplicaciones, informaciones. Pueden ofrecer vistas personalizadas a usuarios específicos. Generalmente en los portales se agrupan la información de interés categorizando el contenido. La programación de portales requiere muchos recursos de cómputos y por el congestionamiento de tráficos de internautas necesitan servidores dedicados.

Los portales tienen dos tendencias: portales horizontales, portales verticales.

Portales horizontales: son masivos, su objetivo es el público en general, aunque pueden tener secciones para usuarios específicos, mayoritariamente de carácter informativo y publicidad.
Portales verticales: son especializados, dirigidos a un público específico con objetivos determinados.

1.1.3 Computación Grid

El término Grid hace referencia a una tecnología innovadora perteneciente al paradigma de computación distribuida, permite compartir recursos computacionales a través de la red. Su tecnología estándar es Globus Toolkit (herramienta libre que hoy en día es el framework por excelencia para desarrollo de Grid middleware).
Es una infraestructura que provee a los usuarios de la potencia de ordenadores que no aprovechan al máximo sus posibilidades los cuales se encuentran dispersos geográficamente, o sea, estos benefician a los usuarios con sus recursos sobrantes sin determinar a qué ordenador conectarse. Además posibilita la comunicación extensa y rápida de forma transparente y segura. En la computación Grid intervienen varias organizaciones cada una con sus recursos de cómputo. Estos recursos se combinan dinámicamente para resolver problemas concretos formando organizaciones virtuales.

1.1.3.1 Grid y la ciencia.

Existen muchas aplicaciones reales que utilizan la Grid, mayoritariamente acotadas en el mundo de la ciencia, dado que son aplicaciones con grandes necesidades computacionales de almacenamiento y procesamiento de datos. Sus usos científicos que se han desarrollado hasta la actualidad están relacionados con la simulación molecular, física de partículas, modelado del clima, observación de la tierra, estudio del genoma humano entre otras.

En el área de la Bioinformática utilizan esta tecnología como alternativa viable a su demanda creciente de potencia de cálculo. Numerosos son los proyectos a nivel mundial que implementan la infraestructura Grid. Ejemplo de ello lo constituyen:

- EGEE: (Enabling GRIDS for e-Science in Europe) cuyo objetivo es desplegar una infraestructura de servicios GRID en Europa disponibles a los científicos 24 horas al día. Actualmente es muy jerarquizada, organizada y controlada que esta desarrollando su middleware propio, gLite. Representa la mayor infraestructura Grid a nivel mundial.

- IRISGrid: Iniciativa que involucra a los principales grupos de investigación de tecnología Grid que hay en España. Tiene como objetivo crear la infraestructura Grid nacional que permita el uso de esta tecnología tanto a nivel de aplicabilidad en diferentes ámbitos. En el caso de la Bioinformático desarrolló un experimento con una aplicación para el cálculo de la predicción de la estructura y las propiedades termodinámicas de una proteína partiendo de su secuencia de aminoácidos.
- BioinfoGRID: el objetivo de este proyecto es conectar varios centros de cómputo presentes en Europa para impulsar el desarrollo de la Bioinformática y las nuevas aplicaciones en esta rama de la ciencia usando una red de servicios basados en la tecnología Grid. El proyecto BioinfoGRID pretende realizar investigaciones en los campos de Genómica, Proteómica y Dinámica Molecular de forma más fácil, reduciendo el tiempo de cálculo para el procesamiento de los datos al distribuir el cómputo entre miles de computadoras de toda Europa. [4]

1.1.3.2 Portales Grid

Los middleware se encargan de exponer los recursos distribuidos de manera transparente, sin embargo el acceso a los mismos no es tan simple. Los portales Grid se encargan de ocultar la infraestructura para acercarla a usuarios no especializados. El portal representa una solución fiable para facilitar a los distintos grupos de usuarios de diferentes ámbitos y disciplinas acceder a los recursos Grid. Constituyen una entrada a una colección de servicios de red distribuidos a los que se puede acceder mediante un navegador.

Los portales Grid evitan que los usuarios tengan que instalar y administrar sus aplicaciones desde sus ordenadores y permite el acceso a la Grid desde cualquier lugar.

Los portales Grid se dividen generalmente en:

Portales orientados al usuario: Son genéricos mayoritariamente definidos para que los usuarios accedan a los recursos Grid.

Portales orientados a la aplicación: Están basados en un diseño especializado donde los usuarios puedan acceder a una determinada aplicación.

Los portales Grid[5] se diseñan e implementan utilizando herramientas estándar y lenguajes de programación basándose en kits de desarrollo. Bibliotecas de funciones, aplicaciones y conjuntos de plantillas que facilitan su elaboración.

Ejemplos de Portales Grid:

- EnterTheGrid: Es el directorio mas extenso de las actividades relacionadas de la Grid. Proporciona una gama de servicios comprensiva. Utiliza las técnicas de gerencia más avanzadas del conocimiento, como MPEG-7 basa clasificaciones, los mapas del asunto de XML y la integración basada XML con otras fuentes de información. Almacena información de proyectos, compañías y productos de todo el mundo. Es mas que una base de datos, es una base de conocimiento basada en XML (xml-based).Proporciona una perspectiva del almacenamiento en-línea y los sistemas de respaldo en-línea.
- GridComputingPlanet: permite a la comunidad técnica actualizarse respecto a los progresos de la Grid.
- GRIDSTART: Iniciativa patrocinada por la Comisión de las Comunidades Europeas con el objetivo específico de consolidar avances técnicos en Europa, de animar la interacción entre actividades similares en Europa y el resto del mundo y de estimular la compensación temprana por industria y la investigación de usos de la Grid permitidos.
- Gridtoday: Colección de noticias y de editoriales diarios de la computación Grid.
- NPACI-Hotpage: Está diseñado para simplificar y consolidar el acceso a los sistemas de cálculo avanzados.
- GridSphere: Proporciona un portal basado en portlet, estándar avanzado de código abierto de la web. Es sofisticado, flexible para personalizar de acuerdo a las necesidades de los usuarios finales. Permite desarrollar y empaquetar portlet que puedan funcionar y ser administrados dentro de este contenedor.

1.1.4 Portlet

Un portlet[6] es un componente Web desarrollado en Java y especificado en el estándar Java Specification Request 168 (JSR168)[7]. El mismo es controlado por un contenedor capaz de generar contenido dinámico. Este contenido no es más que fragmentos de código de lenguaje de marca (HTML, XHTML, WML). Un fragmento se puede agregar a otros fragmentos. Este contenido generado puede variar de un usuario a otro en dependencia del tipo de configuración que haya tenido el portlet. Normalmente, los portlets tienen una clara separación entre el contenido y la presentación, la cual es manejada por una o más clases de Java que contienen la aplicación lógica. De manera general se puede

decir que un portlet no es más que la encapsulación de una aplicación web para poder ser utilizada como un componente dentro de un portal.

1.1.4.1 ¿Cómo funciona un Portlet?

El Portlet tiene un ciclo de vida y según la especificación Java Specification Request 168 (JSR168) está basado en tres fases:

- Inicio (Init): el usuario, al interactuar con el portal arranca el Portlet activando el servicio.
- Gestión de peticiones (Handle requests): procesa la petición mostrando diferentes informaciones y contenido según el tipo de petición. Los datos pueden redimir en sistemas diferentes. Dentro de esta fase se encuentra la Presentación, en la que el Portlet da salida a la información en formato código para su visualización en el navegador.
- Destrucción, (Destroy): elimina el Portlet cuyo servicio deja de estar disponible.

La especificación Java Portlet (JSR 168) permite la interoperabilidad de los portlets entre portales web diferentes. Esta especificación define un conjunto de API para interacción entre el contenedor portlet y el portlet que direcciona áreas de personalización, presentación y seguridad.

1.2 Revisión de las aplicaciones de Servicios Bioinformáticos existentes.

En el mundo existe una gran variedad de aplicaciones que prestan servicios a la Bioinformática desarrolladas en software libre o privado orientadas a: alineamiento de secuencias, identificación de motivos de proteínas incluyendo análisis de dominios, análisis de patrones de secuencias de nucleótidos, análisis de utilización de codones para genomas pequeños, análisis de evolución molecular, análisis del genoma humano utilizando, por ejemplo, la minería de datos.

La mayoría de estos programas trabajan con mucha información y generan enormes cantidades de datos, por lo que exigen una alta demanda de cómputo y no son operables en todo tipo de ordenador. Además si son privados provocan una dependencia tecnológica unida a las costosas patentes. Ante esta limitante, la tendencia mundial en los últimos años es desarrollar aplicaciones web que brindan Servicios Bioinformáticos de modo que utilicen esos recursos a través de la red y no desde sus estaciones de

trabajos. Estas aplicaciones deben estar soportadas por toda una infraestructura de cálculo que permita el procesamiento de las mismas.

A continuación mencionamos algunos ejemplos de aplicaciones web vigentes en la red. Estas se dirigen fundamentalmente hacia un tema en particular (aunque algunas tratan varios) ofreciendo herramientas, base de datos en línea, u algún software de desarrollo para la agilización de las investigaciones científicas y ofrecen bondades propias de los portales.

Phylogeny programs:

Plataforma donde se encuentra la mayoría del software para análisis filogenético (383 paquetes). Programas que cumplen estándares de la calidad y muchos de ellos están difundidos en la web y son libres. Realiza actualizaciones semanalmente y cuanta con mas de 50 servidores libres lo cual facilita la navegación, además permite interacción con el usuario mediante correos.

National Center for Biotechnology Information (NCBI)

El Centro Nacional para la Información Biotecnológica es una rama de los Institutos Nacionales de la Salud en los Estados Unidos que conduce la investigación en la Bioinformática para una mejor comprensión de los procesos moleculares que afectan la salud humana, para ello, desarrolla las herramientas de software que analizan datos del genoma y disemina la información biomédica, las cuales almacena. Crea bases de datos públicas en línea, todas disponibles de forma gratuita. Asigna un identificador único (número de identificación de la taxonomía) a cada especie de organismo. Proporciona además un índice de artículos científicos relacionados con la materia que se manipula. El NCBI es constantemente actualizado.

Malaysian Genomics Resourse Centre (MGRC)

Constituye una plataforma en línea para brindar servicios en la Bioinformática. El centro para las operaciones mundiales reside actualmente en Kuala Lumpur, Malasia. MGRC tiene usuarios por todo el mundo y se ha establecido para proporcionar fácil acceso a las herramientas de análisis ultrarrápidos de la secuencia para la comunidad Bioinformática. Su propósito general es servir a los investigadores y a los científicos individuales que requieren usos del alto-rendimiento, procesamiento y las nuevas plataformas

de la base de datos sin las implicaciones asociadas del recurso. Las áreas de investigación están relacionadas con la droga, genómica comparativa.

The Systems Biology Markup Language (SBML)

Los Sistemas biológicos de lenguaje de marca (SBML) constituyen un formato legible por computadoras para representar los modelos de las redes bioquímicas de la reacción en software. Es aplicable a los modelos del metabolismo, señalización de la célula, y muchas otras. SBML se ha estado desarrollando gracias mid-2000 a una comunidad internacional de analistas de programas informáticos y de usuarios.
Los diagramas gráficos de las redes de reacciones bioquímicas son útiles para la presentación visual a los seres humanos, pero en el nivel de software, un diverso formato es necesario para cuantificar un modelo al punto donde puede ser simulado y analizado. Aquí es donde cobra importancia el portal SBML.

EMBnet

Es la Red Europea de Bioinformática a cargo de mantener bases de datos biológicas y aglutinar a los profesionales en Bioinformática .Presentan un grupo científico de nodos que colaboran a través de Europa, Asia África y Latinoamérica. La maestría combinada de los nodos permite que EMBnet proporcione servicios a la comunidad científica que abarcan en su conjunto más de lo que puede proporcionarse por un solo nodo. Ofrece además, en línea talleres de Bioinformática a diferentes niveles, simposiums conferencias y eventos.

1.3 Justificación a la solución propuesta

Las aplicaciones web mencionadas anteriormente son muy útiles para los que trabajan en la disciplina Bioinformática, puesto que les permite profundizar y avanzar en sus investigaciones, así como acelerar en un tiempo considerable el proceso de las mismas. Además provee a sus clientes de programas, servidores de bases de datos y sistemas de lenguajes que son muy difíciles encontrar en estaciones de trabajo. Estos sitios ofrecen servicios muy buenos, sin embargo, en ellos no se podrían hospedar las aplicaciones desarrolladas por el Grupo de Bioinformática de la UCI, pues están destinados más bien a ofrecer servicios a todos los internautas con necesidades científicas, no a satisfacer los intereses de una institución determinada. Estas aplicaciones no brindan privilegios administrativos y no existe ninguna que permita ser descargada e instalada para utilizar sus cualidades.

Ante el análisis realizado anteriormente y la situación problémica vigente, se decide desarrollar una aplicación web que brinde Servicios Bioinformáticos y permita actualizaciones y adaptaciones en el sitio de acuerdo a las necesidades imperantes en un momento dado. No se pretende suplantar ninguna de las

aplicaciones mencionadas, sino organizar y centralizar todos los sistemas Bioinformáticos (creados y los futuros) que se han estado desarrollado en la universidad en una sola aplicación, así como permitir el acceso a ellos según sus necesidades mediante los servicios que se implementen. Es recomendable utilizar la infraestructura Grid creada en la universidad para aprovechar los recursos computacionales de la misma [8], utilizando para ello un portal Grid.

De los portales Grid mencionados anteriormente EnterTheGrid no se ajusta a las necesidades pues lo que se quiere es acceder a las aplicaciones y utilizarlas, no almacenarlas, el GridComputingPlanet es meramente informativo de los avances de la Grid, al igual que el Gridtoday y los portales GRIDSTART y el NPACI-Hotpage están dirigidos hacia otra línea de desarrollo, Sin embargo el framework GridSphere[9] proporciona un portal web basado portlets de código abierto que no solo realiza las funciones descritas anteriormente sino que permite:

- Instalar en el portal portlets desarrollados a partir de la especificación JSR 168 de Sun Microsystem en un 100 %.
- Soporte para el desarrollo y la integración fáciles de los nuevos portlets.
- La descripción del portal basada en un formato XML flexible para que la interfaz se pueda modificar y ajustar fácilmente a requisitos particulares.
- Permite encapsular la lógica reutilizable de los portlets en servicios que se pueden compartir entre muchos portlets.
- La persistencia de datos proporciona que Hibernate JDO/OQL como sistema gestor de base de datos.
- Portlets básicos del GridSphere: autenticarse, personalización del perfil de usuario, portlets para la administración del portal para la creación de usuarios, de grupos, de la administración del portlet.

1.4 Fundamentación de la metodología de desarrollo del software a utilizar.

Las metodologías de desarrollo se utilizan para estructurar un proyecto determinado y garantizar hasta cierto punto su calidad, eficiencia en el ciclo de vida de desarrollo del software así como lograr un entendimiento entre el cliente, el proveedor del producto y los stakeholder (usuarios finales que interactúan con la aplicación). Muchas son las metodologías se han diseñado para estos fines, cada unas con sus características específicas. Ejemplo de ella lo constituyen RUP y OpenUP.

1.4.1 RUP (Rational Unified Process en inglés o Proceso Unificado de Rational)

Es la metodología que más se utiliza actualmente, constituye un estándar para grandes proyectos orientado a objetos. Su desarrollo es iterativo e incremental, permite el modelado visual, es desarrollada, mantenida y actualizada por Rational. Constituyen un proceso en el que se definen como sus principales elementos: trabajadores, actividades y artefactos.UML es la base del modelamiento visual de RUP. Su ciclo de vida consta de 4 fases:

Inicio: Se establece la oportunidad y alcance el proyecto. Se identifican todas las entidades externas con las que se trata (actores) y se define la interacción a un alto nivel de abstracción.
Elaboración: Se analiza el dominio del problema, se establece una arquitectura base sólida, se desarrolla un plan de proyecto y elimina los elementos de mayor riesgo para el desarrollo exitoso del proyecto.
Construcción: En esta fase todas las componentes restantes se desarrollan e incorporan al producto. Todo es probado en profundidad. El énfasis está en la producción eficiente y no ya en la creación intelectual. Puede hacerse construcción en paralelo, pero esto exige una planificación detallada y una arquitectura muy estable.
Transición: El objetivo es traspasar el software desarrollado a la comunidad de usuarios. Una vez instalado surgirán nuevos elementos que implicarán nuevos desarrollos (ciclos).

RUP proporciona espacios de trabajo seguros para cada desarrollador, suministrando el aislamiento de los cambios hechos en otros espacios de trabajo y controlando los cambios de todos los elementos de software (modelos, código, documentos, etc.).Es muy exigente con mucha documentación y vulnerable a cambios bruscos.

1.4.2 OpenUP (Open Unified Process en inglés o Proceso Unificado Abierto)

Constituye una metodología ágil. Este proceso puede ser desarrollado para hacer frente a una extensa diversidad de tipos de proyectos. Además OpenUP sostiene las mismas características de RUP, contiene el desarrollo iterativo, casos de uso y escenarios de conducción de desarrollo, gestión de riesgos, y el enfoque centrado en la arquitectura. Existe una forma más básica y fácil de manejar de OpenUP que es OpenUP / Basic, objetivos más pequeños y con sede equipos interesados en el desarrollo ágil e iterativo.

OpenUP/Basic no incluye contenido para el despliegue, gestión del cambio, o entorno (tal como personalizar este proceso o preparar el entorno de desarrollo).OpenUP/Basic se enfoca en un equipo único, y estas áreas son tratadas a nivel organizacional o empresarial. Revisar las extensiones de OpenUP que abordan estas áreas más ampliamente.

OpenUP/Basic es un proceso de desarrollo de software que es mínimo, completo y extensible.

Está regido por cuatro principios básicos:

- Prioridades compitiendo por un balance para maximizar el valor para los stakeholders.
- Colaboración para alinear los intereses y un entendimiento compartido.
- Evolucionar para obtener continuamente retroalimentación y mejora.
- Enfoque en articular la arquitectura.

1.4.3 Decisión

Por una parte tenemos RUP, que más tradicional y se centra especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, y las herramientas y notaciones que se usarán. Esta propuesta puede ser efectiva para muchos proyectos, pero para el desarrollo de nuestra aplicación es más factible OpenUP que es para proyectos pequeños de corta duración y mas flexible a los cambios.

1.5 Lenguajes de programación Web.

Son herramientas que nos permiten crear programas y software. Existen múltiples lenguajes sin embargo para implementar nuestra aplicación solo se basa en unos pocos pensados para desarrollar portlet y alguna que otro tipo de implementación: Java Server Pages y algunos lenguajes de marcado como HTML, WML, XHTML.

1.5.1 Java

Java es un lenguaje de programación y una plataforma de desarrollo con el que podemos realizar cualquier tipo de programa. En la actualidad es un lenguaje muy extendido y cada vez cobra más importancia tanto en el ámbito de Internet como en la informática en general. Está desarrollado por la compañía Sun Microsystems con gran dedicación y siempre enfocado a cubrir las necesidades tecnológicas más punteras.

Una de las principales características por las que Java se ha hecho muy famoso es que es un lenguaje independiente de la plataforma. Eso quiere decir que si hacemos un programa en Java podrá funcionar en cualquier ordenador del mercado. Es una ventaja significativa para los desarrolladores de software, pues antes tenían que hacer un programa para cada sistema operativo, por ejemplo Windows, Linux, Apple.

Esto lo consigue porque se ha creado una Máquina de Java para cada sistema que hace de puente entre el sistema operativo y el programa de Java y posibilita que este último se entienda perfectamente.

La independencia de plataforma es una de las razones por las que Java es interesante para Internet, ya que muchas personas deben tener acceso con ordenadores distintos. Pero no se queda ahí, Java está desarrollándose incluso para distintos tipos de dispositivos además del ordenador como móviles, agendas y en general para cualquier cosa que se le ocurra a la industria.

1.5.2 Java Server Pages

JSP es un acrónimo de Java Server Pages. Es una tecnología orientada a crear páginas web que generan contenido dinámico con programación en Java. Con JSP podemos crear aplicaciones web que se ejecuten en variados servidores web, y es en esencia un lenguaje multiplataforma. Las páginas JSP están compuestas de código HTML/XML mezclado con etiquetas especiales para programar scripts de servidor en sintaxis Java. Por tanto, las JSP podremos escribirlas con nuestro editor HTML/XML habitual.

1.5.3 HTML

El Lenguaje para el Formato de Hipertextos es la lengua franca de la World Wide Web, se utiliza en la creación de páginas web y otros tipos de documentos que habitualmente son visualizados con un navegador.

La última versión del HTML es la 4.01. Posteriormente fue reformulado en XML dando lugar al XHTML, la versión actual es la 1.1.

HTML y XHTML son estándares del W3C.

1.6 Herramientas de desarrollo

Para desarrollar nuestro proyecto se utilizan las siguientes herramientas.

1.6.1 Eclipse

Eclipse es un proyecto de desarrollo de software de código fuente abierto (*open source*), la mayor parte del cual ha sido escrito en Java. Cuyo objetivo es la construcción de herramientas integradas para el desarrollo de aplicaciones.

1.6.2 Herramienta Case

Dado que la metodología por la cual se va a regir el presente trabajo es OpenUP la herramienta case a utilizar es Visual Paradigm. Herramienta UML profesional que soporta el ciclo de vida completo de desarrollo de software: Análisis y diseño orientado a objetos, construcción, prueba y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y aun menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación.

1.7 Patrones de Arquitectura y de Diseño

1.7.1 Patron de Arquitectura .Modelo Vista Controlador

En la actualidad uno de los patrones más utilizados para cualquier tipo de aplicaciones son los arquitectónicos. El mundo Java adoptó el patrón Modelo Vista Controlador (MVC) para las aplicaciones Web como patrón de arquitectura. [10]

Este patrón de arquitectura de software separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón MVC se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página, el modelo es el Sistema de Gestión de Base de Datos y el controlador representa la Lógica de negocio.

Descripción del patrón

- **Modelo:** Administra el comportamiento y los datos del dominio de aplicación, responde a requerimientos de información sobre su estado (usualmente formulados desde la vista) y responde a instrucciones de cambiar el estado (habitualmente desde el controlador).

- **Vista:** Maneja la visualización de la información y presenta el modelo en un formato adecuado para interactuar, usualmente un elemento de interfaz de usuario (muestra información del modelo de usuario).

- **Controlador:** Controla el flujo entre la vista y el modelo (los datos). Responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

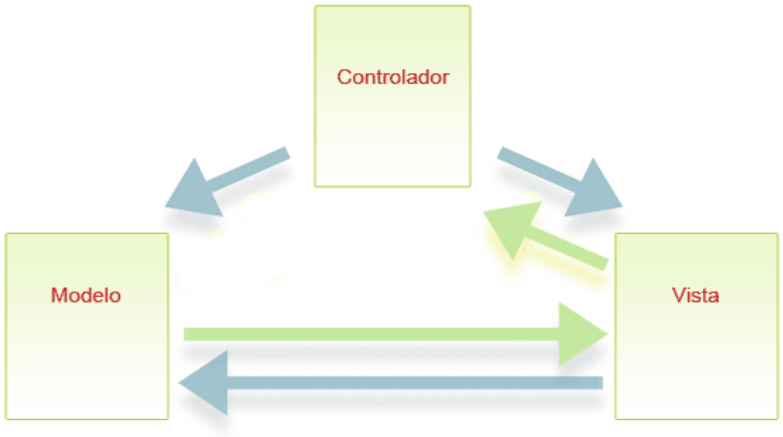


Figura 1 Patrón Modelo Vista Controlador.

1.7.2 Patrón de Diseño.

Un patrón es un modelo, una solución, que podemos aplicar o seguir para realizar algo. Estos surgen de la experiencia de personas al tratar de lograr ciertos objetivos. Los patrones capturan la experiencia existente y probada para promover buenas prácticas.

"Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular." [11]

Patrón GRASP:

Para obtener mayor calidad en el diseño se tuvieron en cuenta una serie de patrones como son los patrones GRASP (Patrones generales de software para asignar responsabilidades). Las responsabilidades están relacionadas con las obligaciones de un objeto en cuanto a su comportamiento. Estas responsabilidades son básicamente de dos tipos: [12]

Conocer:

Conocer los datos privados encapsulados.

Conocer los objetos relacionados.

Conocer las cosas que puede derivar o calcular.

Hacer:

Hacer algo él mismo, como crear un objeto o hacer un cálculo.

Iniciar una acción en otros objetos.

Controlar y coordinar actividades en otros objetos.

En la aplicación se emplearon los siguientes:

- **Bajo Acoplamiento:** Cada clase está acoplada solamente a las clases necesarias.
- **Alta Cohesión:** Se asignaron responsabilidades a las clases de manera que todos sus métodos tengan un comportamiento bien definido.

Conclusiones

De acuerdo al análisis del estado del arte de las aplicaciones de Servicios Bioinformáticos realizado en el presente capítulo, apoyado en los conceptos previamente definidos y teniendo en cuenta la situación problémica, se decide desarrollar un portal Grid de carácter vertical , para ello se escoge el Gridsphere como framework que satisface los intereses del Grupo de Bioinformática de la UCI. Se utiliza como metodología Open Up y como herramienta Case Visual Paradigm. Se definieron además los patrones y herramientas a utilizar en el desarrollo de la aplicación.

Capitulo 2 Características del Sistema

Introducción.

En el presente capítulo se hace un análisis de la propuesta del sistema, para ello se especifican los requisitos funcionales y no funcionales, se identifica mediante el diagrama de casos de uso del sistema las relaciones entre los actores y casos de uso del sistema; así como las descripciones textuales de cada caso de uso.

2.1 Requisitos Funcionales.

Los requerimientos funcionales son aquellos que responden a: ¿qué debe hacer el sistema? y describen las capacidades que el sistema debe cumplir.

El sistema propuesto tiene los siguientes requisitos funcionales:

R1 Gestionar portlet.

- **R1. 1** Adicionar portlet.
- **R1. 2** Modificar portlet.
- **R1. 3** Eliminar portlet.

R2 Autenticar usuario.

R3 Selecionar portlet

R4 Cargar Moleculas.

R5 Convertir Moléculas.

R6 Editar perfil.

R7 Descargar Moleculas.

2.2 Requisitos no Funcionales.

Los requerimientos no funcionales son las propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. En muchos casos los requisitos no funcionales son fundamentales en el éxito del producto. Normalmente están vinculados a requerimientos funcionales, es decir una vez se conozca lo que el sistema debe hacer podemos determinar cómo ha de comportarse, qué cualidades debe tener o cuán rápido o grande debe ser.

Para el sistema propuesto se han definido los siguientes requisitos no funcionales:

Seguridad:

Los usuarios necesariamente deberán estar autenticados para acceder a muchos de los contenidos y servicios que brinda el portal, sin embargo los administradores se comportarán como usuarios, pero para poder además realizar configuraciones de los contenidos, modificar y realizar servicios de la aplicación referente a los roles y permisos asignados, si tendrán obligatoriamente que estar autenticados. La aplicación contará con protección contra acciones no autorizadas y que puedan afectar la integridad de los datos. Deberá estar disponible las 24 horas del día los 7 días de la semana garantizando un acceso de forma fácil y rápida para los usuarios.

Rendimiento:

El sistema al estar concebido para un ambiente cliente/servidor, tratará de garantizar la rapidez de respuesta ante las solicitudes de los usuarios, al igual que la velocidad de procesamiento de la información.

Usabilidad:

La aplicación tendrá un ambiente sencillo y será fácil de manejar para los usuarios incluso aquellos que no han tenido mucha experiencia en el trabajo con computadoras o con sistemas informáticos.

Apariencia o interfaz externa:

La aplicación debe contar con una interfaz amigable que le permita al usuario interactuar de forma cómoda con la misma y le agilice y facilite el trabajo con el software. La página principal tendrá información que le servirá de guía al usuario. Debe ser preferentemente de color verde que muestre el logo del proyecto.

Portabilidad:

El sistema podrá ser ejecutado sobre los sistemas operativos Linux y Windows, por su característica de ser multiplataforma, de forma que permita una fácil migración haciendo uso de estándares y tecnologías de código abierto.

Soporte:

Una vez terminada la aplicación y realizadas las pruebas piloto del software. Si está lista para comenzar su ejecución, se pondrá a la disposición de los usuarios donde estos tendrán la posibilidad de acceder y así lograr evolutivamente la integración con la aplicación. El software será actualizado sistemáticamente por los administradores del sistema; también será garantizado su mantenimiento.

Software:

Se debe disponer para poder instalar la aplicación:

En el cliente:

Sistema operativo GNU/Linux o Windows 98 o superior.

Navegador Web estándar con capacidad de interpretación de Java Script, CSS y Ajax compatible con la W3C ejemplo Netscape, optimizado para Mozilla 1.7 o superior o FireFox 0.9.3 o superior.

En el servidor:

Servidor Web Apache Tomcat.

Java 5 EE

Hardware

En el caso de las PC servidoras deberán contar con un microprocesador Intel Pentium III o superior, a 450 MHz o superior, con un mínimo de 256MB de memoria RAM, un disco duro con capacidad de 40GB y una tarjeta de red alámbrica o inalámbrica, o módem.

En el caso de las estaciones de trabajo o de usuarios se necesitan PC con microprocesador Intel Pentium II o superior, con un mínimo de de 128MB de memoria RAM, el disco duro con capacidad de 4 GB como mínimo, monitor con resolución de 800x600 o superior y tarjeta de red alámbrica o inalámbrica para la conexión de red local, o módems para otro tipo de conexión.

Legales

Se usarán herramientas de software libre y código abierto, o que funcionen bajo sistemas operativos representativos de código abierto, bajo las licencias GNU/GPL.

Restricciones en el diseño y la implementación

El análisis y diseño de la aplicación será basado en la Metodología Open Up con el uso del lenguaje de modelado UML. Se usará como herramienta CASE Visual Paradigm para el modelado. Se usará como lenguaje de programación Java (JSP).

2.3 Descripción del Sistema Propuesto.

El portal debe permitir el acceso de los usuarios o personas a los diferentes servicios en dependencia del rol que desempeñen dentro de este.

Si la persona que entra al sistema es un usuario que nunca ha sido registrado solo podrá visualizar los contenidos del portal.

La persona o usuario que decida registrarse una vez que haya entrado al portal podrá hacer lo mismo que el usuario que no se registra, pero además puede seleccionar portlet que va a utilizar, podrá cargar, convertir, descargar las moléculas del portlet y editar su perfil .El usuario registrado que una vez autentificado se compruebe que es personal autorizado de la facultad con los privilegios concedidos por el administrador, podrá gestionar en si los portlets.

El sistema contará con un administrador que es la persona que tiene todos los privilegios en la aplicación. Está autorizado a hacer cambios informáticos en el portal, gestionar su configuración, controlar toda la aplicación y el mantenimiento del mismo.

2.4 Definición de los Actores del Sistema.

Los actores representan terceros fuera del sistema que interactúan con él. Cada actor juega un rol determinado al interactuar con el sistema y diferentes usuarios pueden asumir el mismo rol de un actor.

Tabla 2.1: Descripción de los Actores del Sistema.

Actor	Descripción
Administrador	Representa al editor general del sistema, quien además gestiona la configuración del mismo y tiene todos los privilegios en la aplicación.
Usuario Autenticado	Representa a los usuarios que ingresan en el sistema.

2.5 Definición de los Casos de Uso del Sistema.

Los casos de uso son fragmentos de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores.

2.5.1 Patrones de Casos de Uso.

“La experiencia en la utilización de casos de uso ha evolucionado en un conjunto de patrones que permiten con más precisión reflejar los requisitos reales, haciendo más fácil el trabajo con los sistemas, y mucho más simple su mantenimiento. Dado un contexto y un problema a resolver, estas técnicas han mostrado ser la solución adoptada en la comunidad del desarrollo de software. Se presentan a modo de herramientas que permiten resolver los problemas que se les planteen a los desarrolladores de una forma ágil y sistemática. Estos patrones se enfocan hacia el diseño y las técnicas utilizadas en modelos de alta calidad, y no en cómo modelar usos específicos.” [13]

Dentro de los patrones utilizados se encuentran:

CRUD (Creating, Reading, Updating, Deleting).

"Este patrón consta de un caso de uso, llamado Información CRUD o Gestionar información, modela todas las operaciones que pueden ser realizadas sobre una parte de la información de un tipo específico, tales como creación, lectura, actualización y eliminación. Suele ser utilizado cuando todos los flujos contribuyen al mismo valor del negocio, y estos a su vez son cortos y simples." [14]

Listado de Casos de Uso.

CU-1 Gestionar portlet.

CU-2 Autenticar usuario.

CU-3 Seleccionar portlet.

CU-4 Cargar Moléculas.

CU-5 Convertir Moléculas.

CU-6 Editar perfil.

CU-7 Descargar Moléculas.

2.6 Diagrama de Caso de Uso del Sistema.

El diagrama de casos de uso del sistema nos ayuda a comprender gráficamente los procesos del sistema y su interacción con los actores. Estos diagramas se agruparon en paquetes que responden a uno de los criterios para el empaquetamiento de los casos de uso del sistema:

- Los CU requeridos para dar soporte a un determinado actor del sistema.

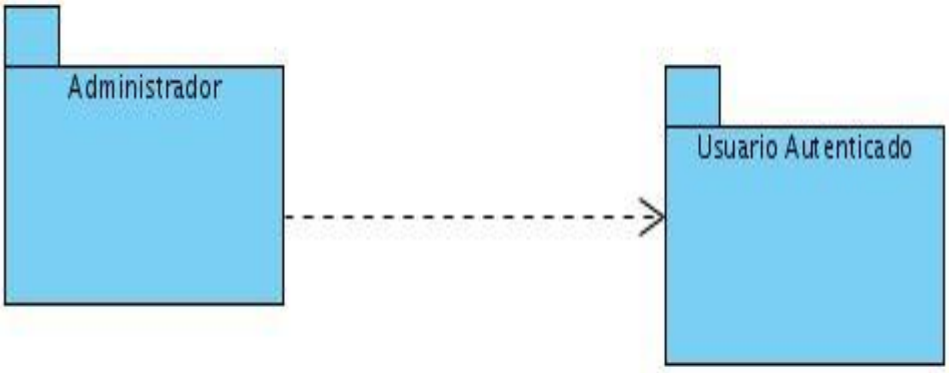


Figura 2 Diagrama de Paquetes del Sistema.

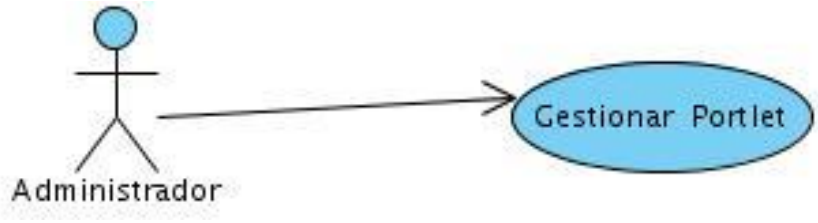


Figura 3 Diagrama de CUS correspondiente al paquete Administrador.

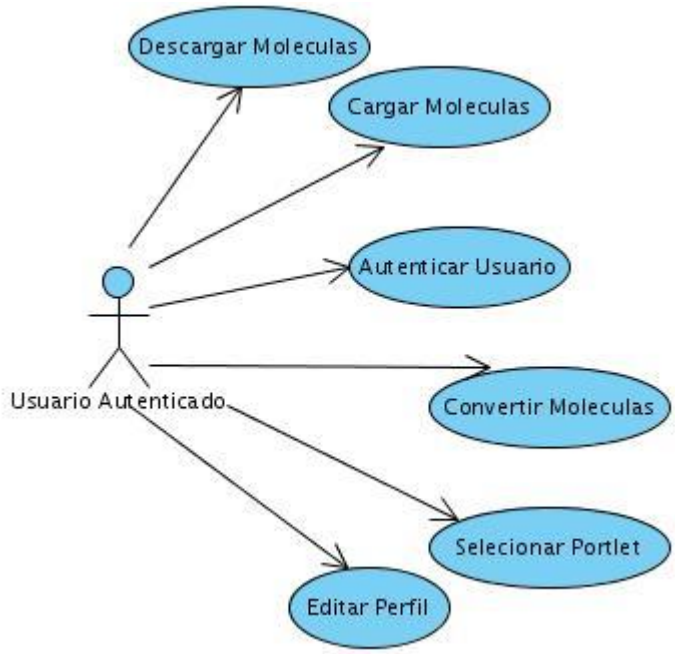


Figura 4 Diagrama de CUS correspondiente al paquete Usuario Autenticado.

2.7 Descripción de los Casos de Uso del Sistema.

Mediante la descripción de los casos de uso del sistema se detalla la secuencia de eventos que los actores llevan a cabo para completar un proceso a través de la aplicación.

Tabla 2.2: Descripción del caso de uso Cargar Moléculas.

Caso de Uso	
CU-4	Cargar Moléculas.
Actores	Usuario Autenticado (inicia).
Propósito	Permitir que se pueda introducir la molécula a convertir.
Resumen	El caso de uso inicia cuando el Usuario Autenticado necesita introducir las moléculas que desea que sea convertida. El sistema muestra la interfaz con los campos donde se debe especificar el formato de molécula de entrada y el de salida. El Usuario Autenticado introduce los datos necesarios. El sistema carga esas moléculas. El caso de uso finaliza cuando el sistema realiza las operaciones satisfactoriamente.

Tabla 2.3: Descripción del caso de uso Convertir Moléculas

Caso de Uso	
CU-5	Convertir Moléculas.
Actores	Usuario Autenticado (inicia).
Propósito	Realizar la conversión de la molécula introducida
Resumen	El caso de uso inicia cuando el Usuario Autenticado desea ver la molécula ya convertida. El sistema convierte internamente la molécula que fue previamente introducida. El caso de uso finaliza cuando el sistema le muestra al Usuario Autenticado la molécula ya convertida

Tabla 2.4: Descripción del caso de uso Descargar Moléculas

Caso de Uso	
CU-7	Descargar Moléculas.
Actores	Usuario Autenticado (inicia).
Propósito	Realizar la descarga de la molécula introducida
Resumen	El caso de uso inicia cuando el Usuario Autenticado desea descargar las moléculas ya convertidas .El caso de uso finaliza cuando el sistema descarga las moléculas.

2.8 Vista de Casos de Uso.

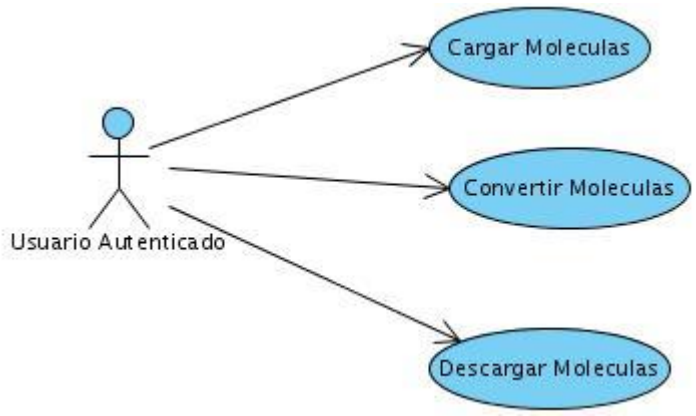


Figura 5 Casos de uso arquitectónicamente significativos.

Conclusiones

En este capitulo quedaron definidos los requerimientos funcionales y no funcionales que debe tener la aplicación, actores y casos de uso que participan en el sistema. Se realizaron los diagramas de casos de uso del sistema, mostrando la relación entre los casos de uso y los actores del sistema. Se describieron los tres casos de uso que fueron programados, los cuales son los casos de uso arquitectónicamente significativos.

Capítulo 3 Diseño del Sistema

Introducción.

En el presente capítulo se modelan los artefactos correspondientes al flujo de análisis y diseño. Se describe la arquitectura del sistema. Se definen los patrones de diseño a utilizar. Se describe el modelo de diseño mediante diagramas de clases del diseño con estereotipos web. También en el mismo esta presente el modelo de despliegue.

3.1 Arquitectura del Sistema.

Patrones arquitectura

MVC

El mundo Java adoptó el patrón MVC para las aplicaciones Web

- La capa de presentación (Vista) está formada por JSPs. Aquí podemos encontrar los Portlet con todos sus componentes, que son las clases con las que interactúa directamente el usuario desde el portal de servicio.
- El controlador suele implementarse como uno o varios Servlet que son las clases encargadas de recibir las diferentes peticiones de los usuarios y hacer llamadas a las funciones necesarias para dar cumplimientos a estas peticiones.
- El modelo de datos puede ser implementado a través de componentes que representan objetos en la base de datos (EJBs (Enterprise Java Beans) o POJOs (acrónimo de Plain Old Java Object) con Hibernate, etc.)

Arquitectura de Portlet.

Se implementa el patrón de arquitectura MVC (Model View Controller) para el desarrollo de portlet. Contiene un Objeto Controller implementado como Portlet parametrizable desde un archivo XML que es el responsable de controlar el flujo de la aplicación, instanciar los beans que sean necesarios y servir los JSP (Vista) que estén activos en cada área. Controla los estados por los que pasa la aplicación desde que un usuario genera un evento, hasta que se devuelve el contenedor asociado.

La Vista es la capa en la que se compone la presentación o página que se va a mostrar al usuario como resultado de su petición. La capa de datos es donde se almacena de forma persistente toda la información necesaria para facilitar los servicios ofrecidos por el portal.

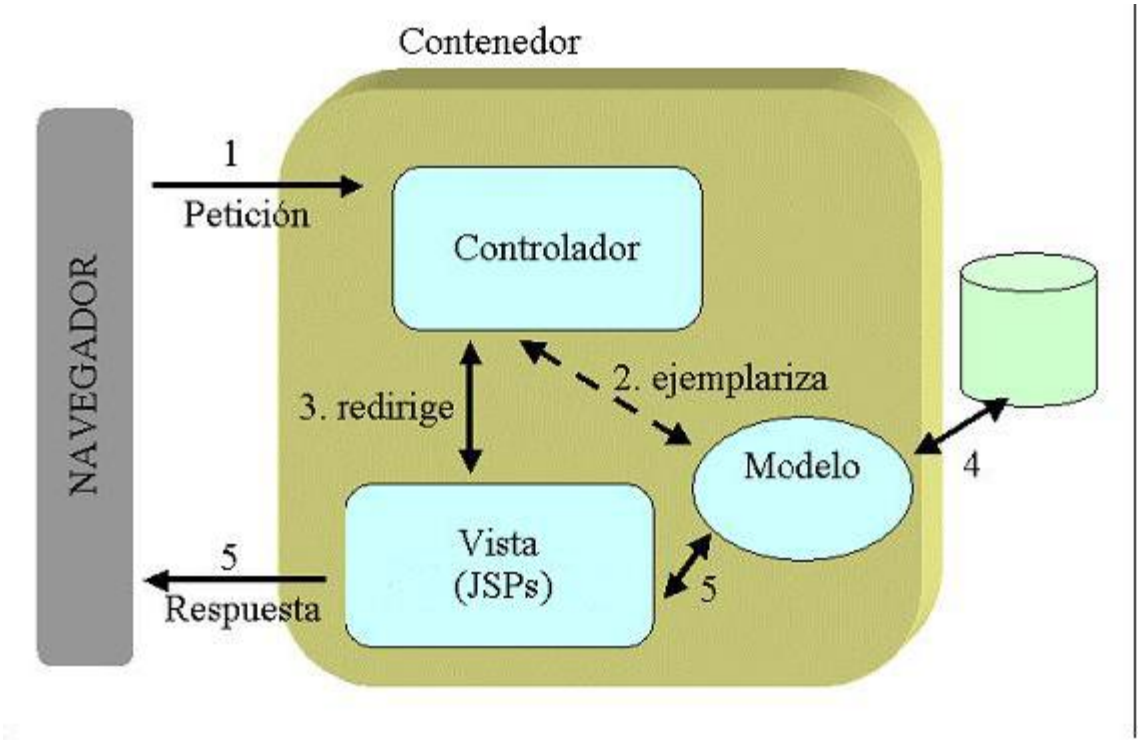


Figura 6 Modelo Vista Controlador.

3.1.1 Vista Lógica.

Diagrama de Clases de Alto Nivel.

En la siguiente figura se pone de manifiesto los tres casos de uso que son programados en nuestro sistema, por lo que podemos decir que son nuestros casos de uso arquitectónicamente significativos. En nuestro capítulo 2 se hace la descripción textual de estos casos de uso.

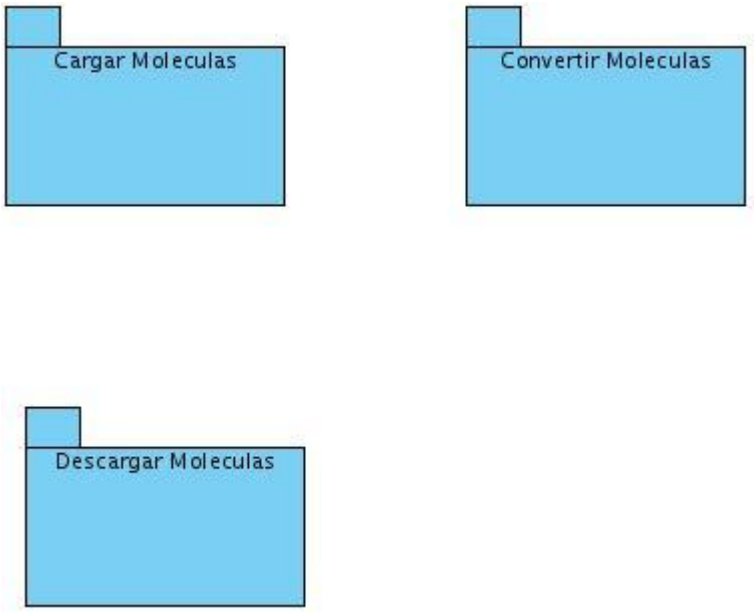


Figura 7 Diagrama de paquetes del diseño.

3.1.2 Vista de Despliegue.

Diagrama de Despliegue.

El siguiente diagrama muestra la configuración hardware del sistema y los nodos físicos que la componen. La aplicación necesita para su ejecución un servidor Web Apache, un servidor de bases de datos HSQLDB, un servidor de aplicaciones, las terminales Web donde cada usuario tendrá acceso a la aplicación.

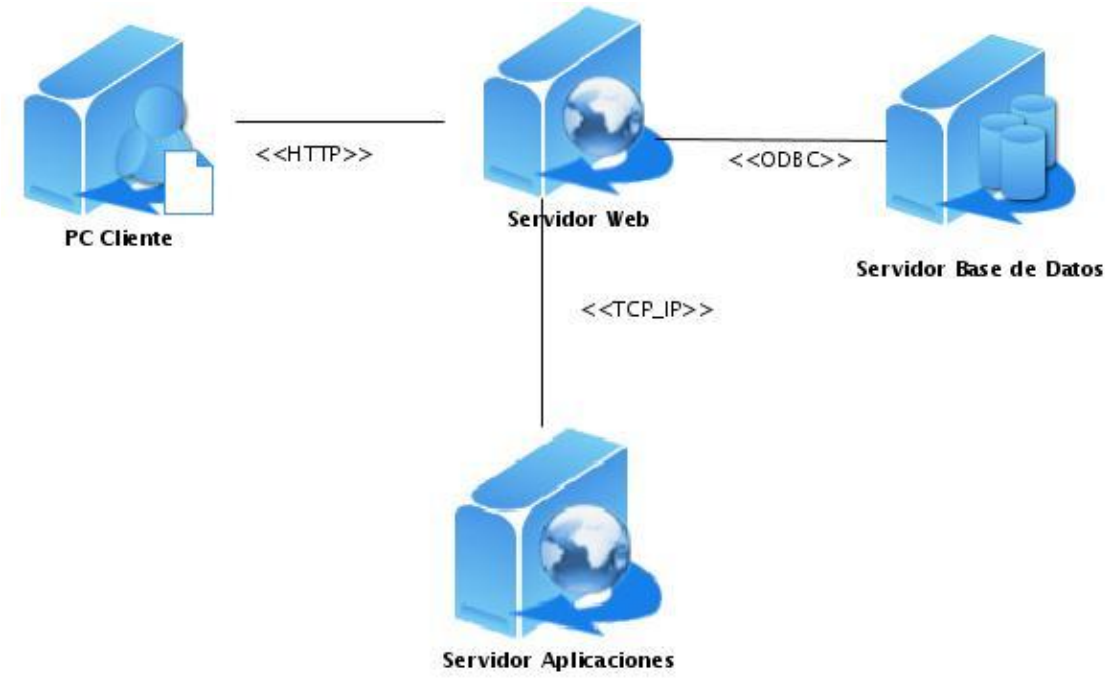


Figura 8 Diagrama de Despliegue.

3.2 Realización de Casos de Uso del Diseño.

3.2.1 Patrones de Diseño.

Lo patrones de GRASP, no compiten con los patrones de diseño… los patrones de GRASP, nos guían para ayudarnos a encontrar los patrones de diseño (que son más concretos).....

Tabla 3.1: Descripción de los patrones GRASP.

Patrón	Descripción
Bajo Acoplamiento	Debe haber pocas dependencias entre las clases. Si todas las clases dependen de todas ¿cuanto software podemos extraer de un modo independiente y reutilizarlo en otro proyecto? Para determinar el nivel de acoplamiento de clases, son muy buenos los diagramas de colaboración de UML. ---------- Uno de los principales síntomas de un mal diseño y alto acoplamiento es una herencia muy profunda. Siempre hay que considerar las ventajas de la delegación respecto de la herencia. Como ejemplo (de mal diseño), en muchos diseños Web se puede ver como se crea un servlet base con capacidades de paginación y se hereda de él para construir los demás. La paginación es un servicio que se podría usar en aplicaciones no Web, por lo que es más adecuado mantener estas capacidades en clases externas. Otro ejemplo clásico se produce cuando se pasan los objetos

Alta Cohesión	relacionados con la capa de presentación a la capa de negocio (HttpRequest, HttpResponse). Cada elemento de nuestro diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable. ---------- Ejemplos de una baja cohesión son clases que hacen demasiadas cosas. En todas las metodologías se considera la refactorización. Uno de los elemento a refactorizar son las clases saturadas de métodos. Ejemplos de buen diseño se producen cuando se crean los denominados "paquetes de servicio" o clases agrupadas por funcionalidades que son fácilmente reutilizables (bien por uso directo o por herencia). ---------- En java, pensar en interfaces nos fuerza a que nuestros sistemas sigan los principios de alta cohesión. En algún sitio leí que una aplicación con menos de 8 interfaces no utiliza correctamente los conceptos de orientación a objeto... y estoy de acuerdo....

Problema: Que debe haber pocas dependencias entre las clases....

Solución: Para dar solución a esto se aplica el patrón Bajo acoplamiento de los GRASP.

Aplicación: Existe una mínima dependencia entre las clases, puesto que la clase convertirSerlevt solo utiliza los ficheros que devuelve la clase ficheroSerlevt y la clase descargarSerlevt requiere nada más el fichero que devuelve convertirSerlevt , o sea la implementación de cada una de ellas no afecta a la otra , solo tiene que asegurarse de que sea la adecuada.

Problema: Que se le asignen responsabilidades a las clases de manera que todos sus métodos tengan un comportamiento bien definido.....

Solución: Para dar solución a esto se aplica el patrón Alta Cohesión de los GRASP.

Aplicación: En nuestro sistema cada clase realiza únicamente la labor que le corresponde, ejemplo:

La clase ficheroSerlevt se encarga de subir los ficheros del cliente al servidor, la clase convertirSerlevt se encarga solo de convertir los datos de las moléculas en el formato indicado por el usuario y la clase descargarSerlevt se encarga de descargar las moléculas convertidas del servidor al cliente.

3.2.2 Diagrama de Clases del Diseño con Estereotipos Web.

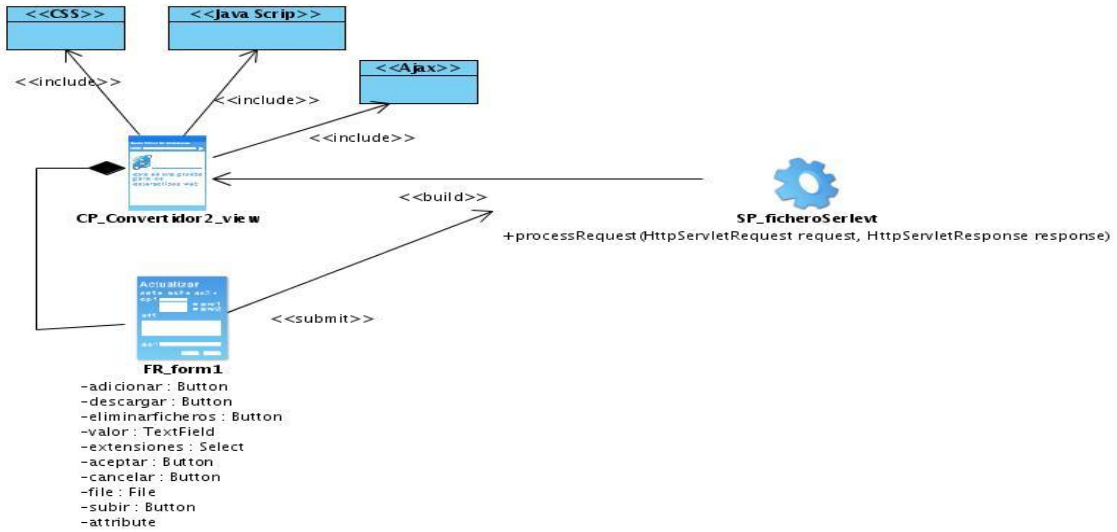


Figura 9 CU-Cargar Moléculas.

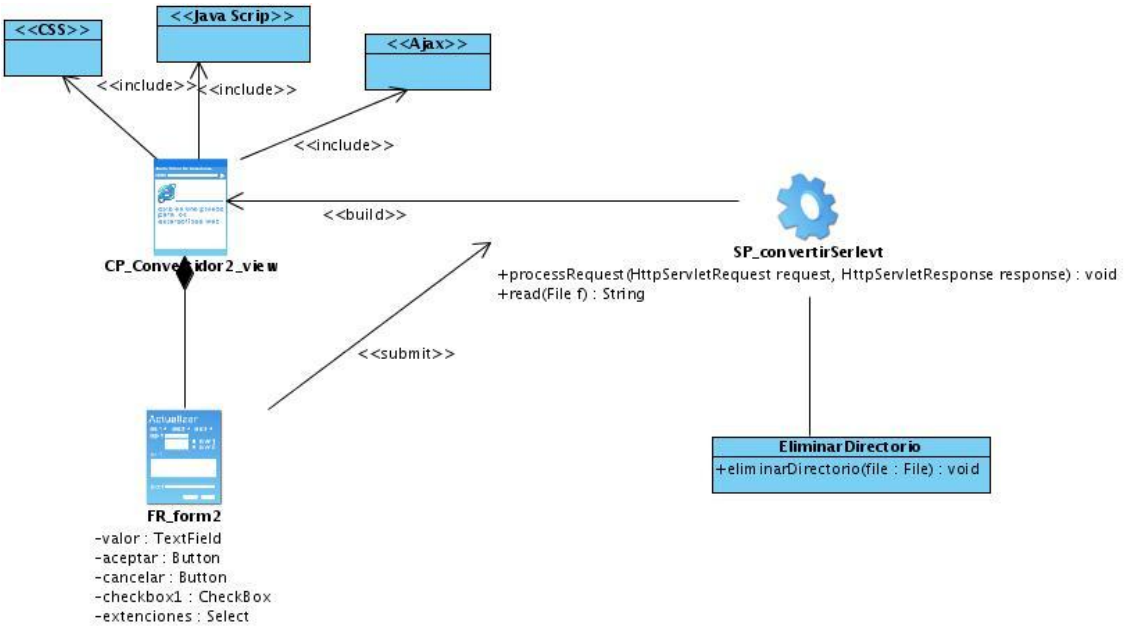


Figura 10 CU-Convertir Moléculas.

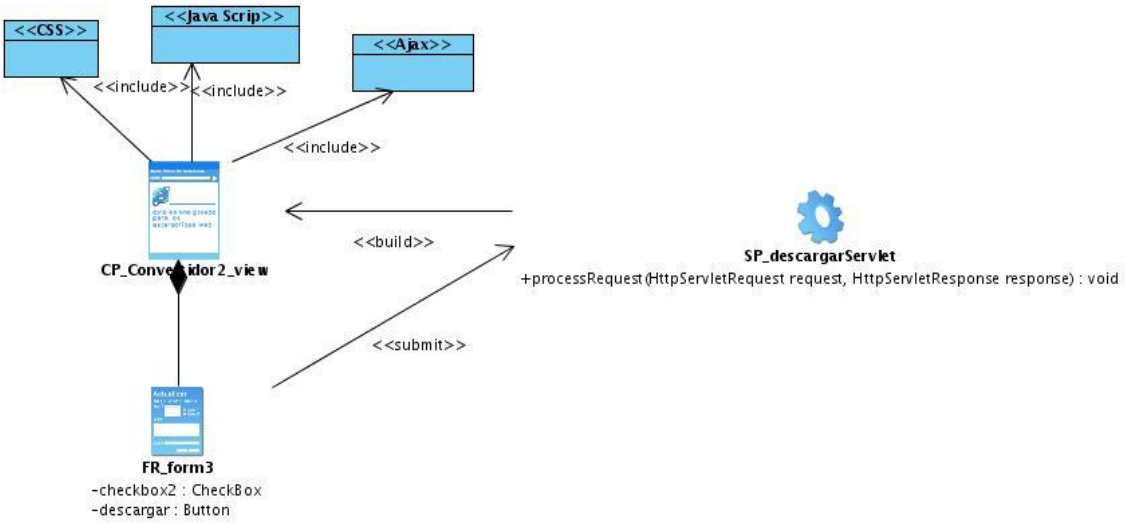


Figura 11 CU_Descargar Moléculas.

3.2.3 Diagrama de Interacción (Secuencia).

Como su nombre indica, estos diagramas muestran una interacción, que consiste en un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar entre ellos. Los diagramas de interacción se utilizan para modelar los aspectos dinámicos del sistema. Se clasifican en Diagramas de Secuencia y Diagramas de Colaboración, los primeros destacan la ordenación temporal de los mensajes y los segundos la organización estructural de los objetos que envían y reciben mensajes.

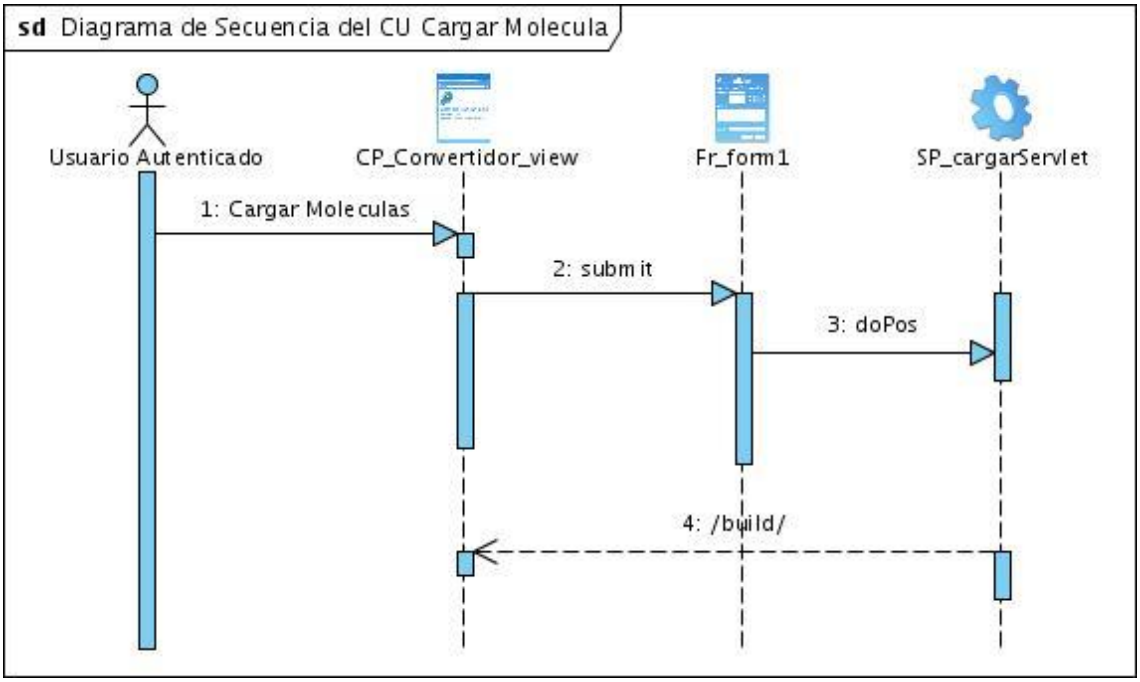


Figura 12 Diagrama de Secuencia del CU Cargar Molécula.

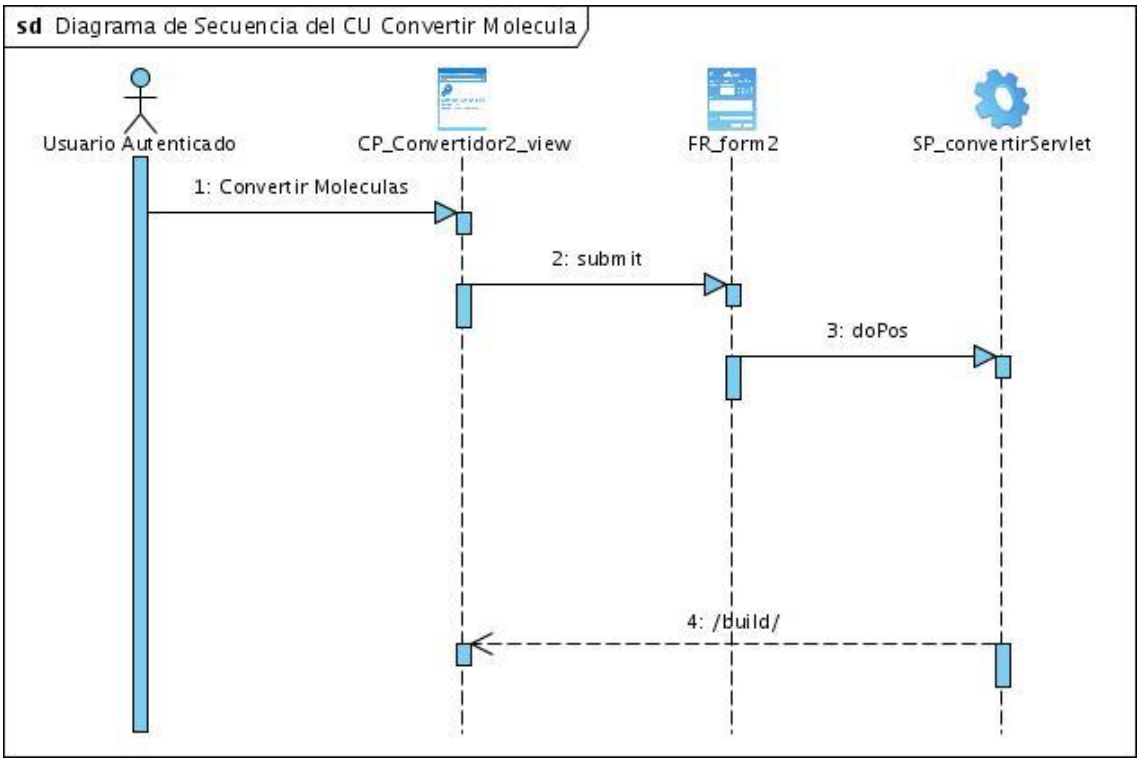


Figura 13 Diagrama de Secuencia del CU Convertir Molécula.

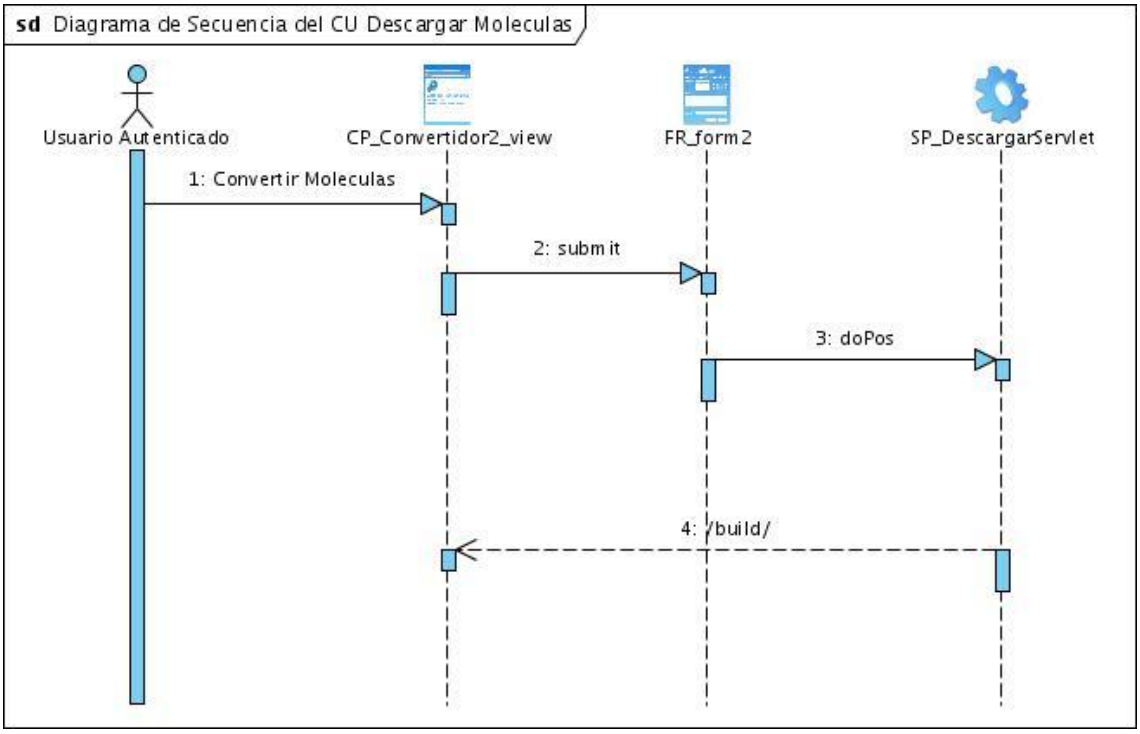


Figura 14 Diagrama de Secuencia del CU Descargar Molécula.

3.3 Principios de Protección y Seguridad.

Para garantizar la seguridad de la información en la aplicación, se crearon varios niveles de seguridad, definidos como tipos de usuarios, teniendo en cuenta los roles que desempeñan los usuarios que interactúan con el sistema. El administrador es la persona que tiene el control total del sistema y es el encargado de crear los usuarios con sus respectivos roles.

Conclusiones.

En este capítulo quedó definida la arquitectura del sistema, se especificaron las clases que contendrán las capas de presentación, negocio y datos. Se muestran los patrones de diseño aplicados. Se realizaron los diagramas de clases del diseño, los diagramas de secuencia donde se modelaron los aspectos dinámicos de la aplicación, el diagrama de despliegue.

Capítulo 4 Implementación del Sistema

Introducción

El objetivo principal de este capítulo es convertir los elementos del diseño en elementos de implementación. Se describe cómo los elementos del modelo del diseño se implementan en términos de componentes. También se presenta el código fuente de las principales clases y pantallas de la aplicación como por ejemplo prototipos de interfaz de usuario y el mapa de navegación.

4.1 Diagramas de componentes

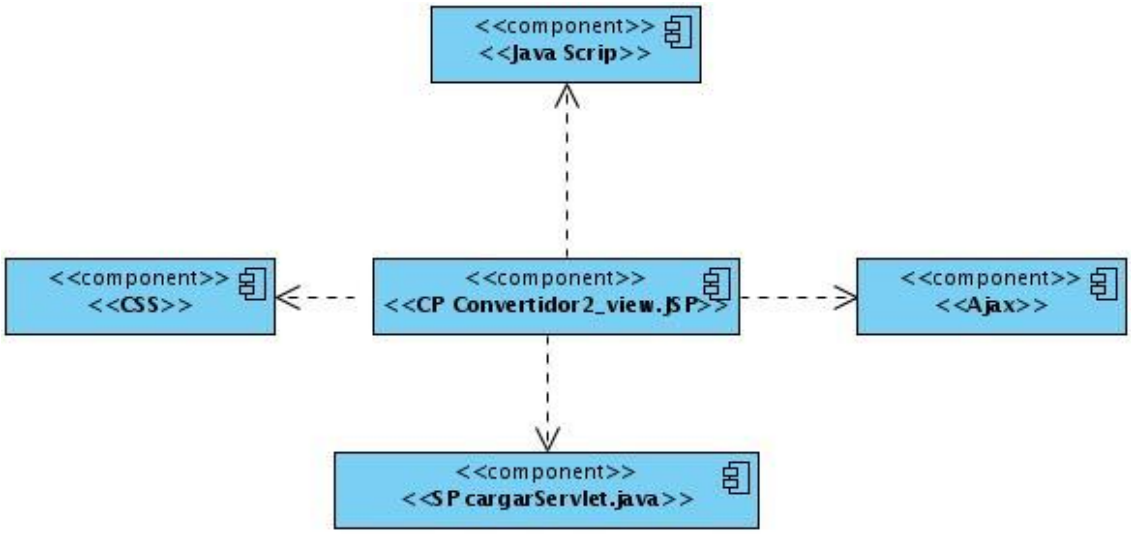


Figura 15 Diagrama de componentes Cargar Moléculas.

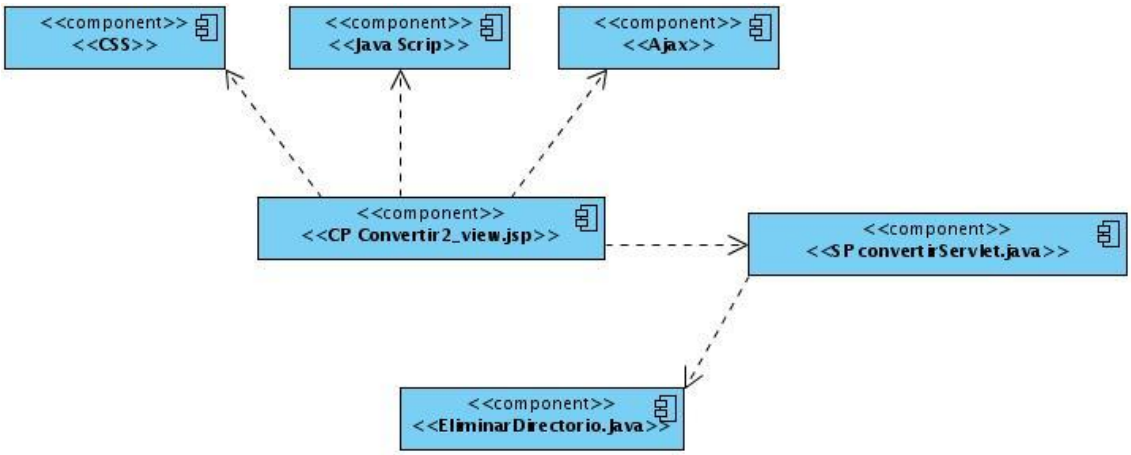


Figura 16 Diagrama de componentes Convertir Moléculas.

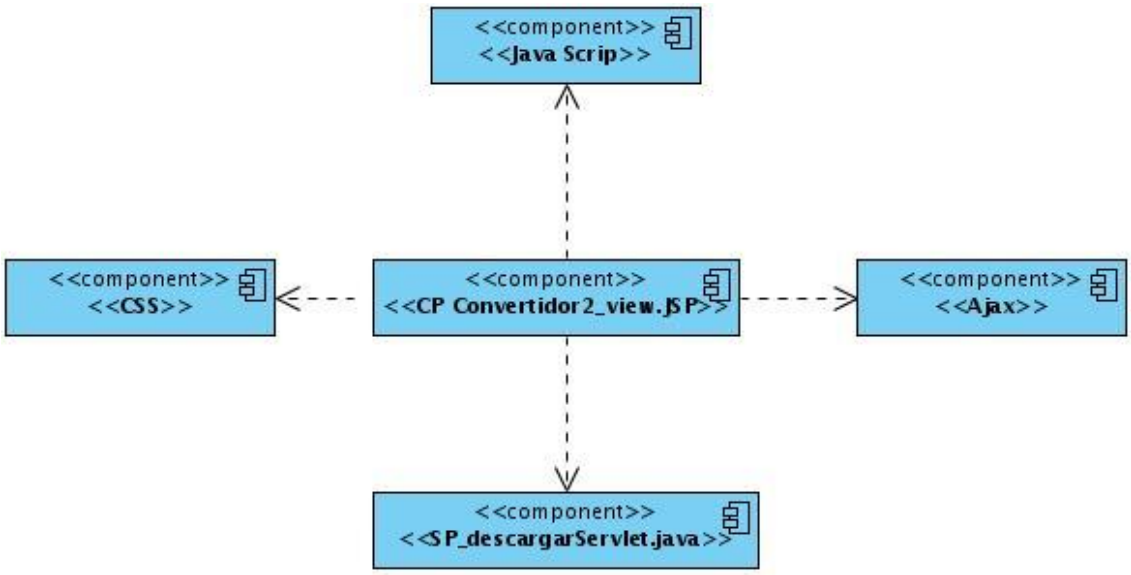
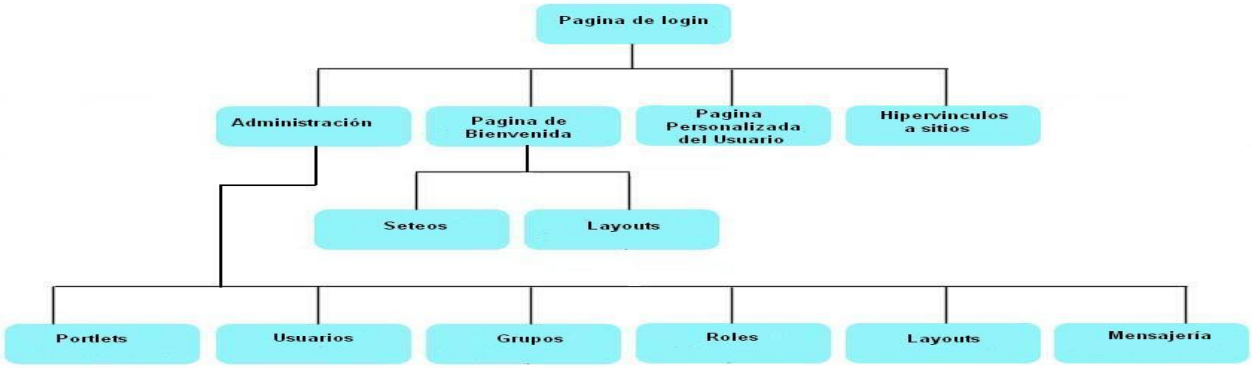


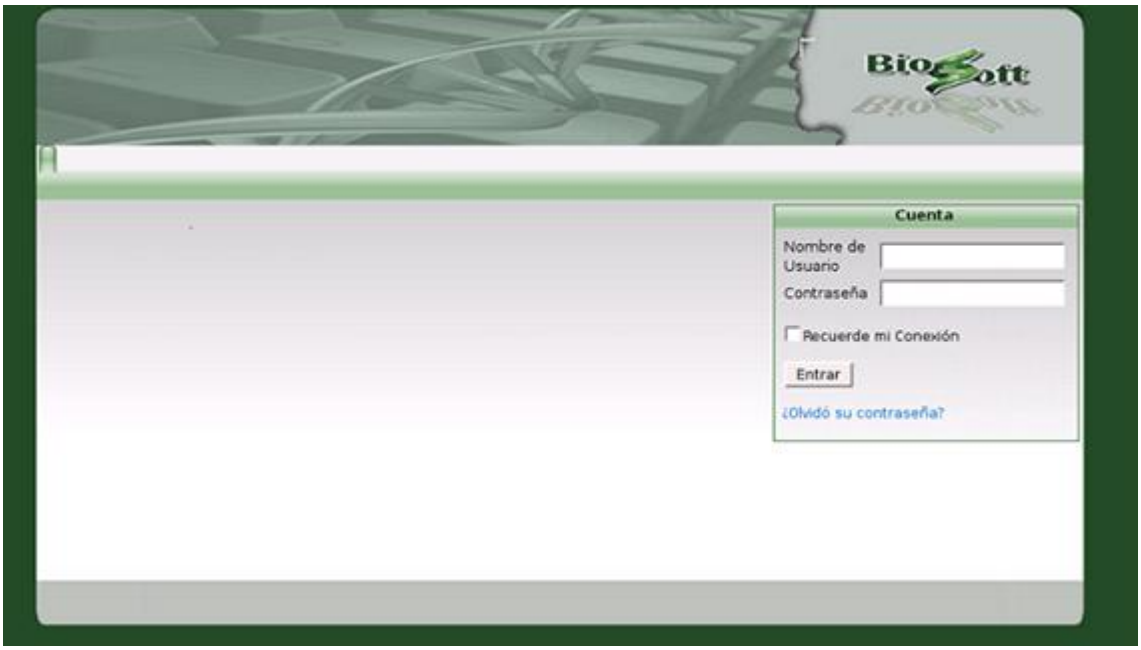
Figura 17 Diagrama de componentes Descargar Moléculas.

4.2 Pantallas de la aplicación.

4.2.1 Mapa de Navegación.

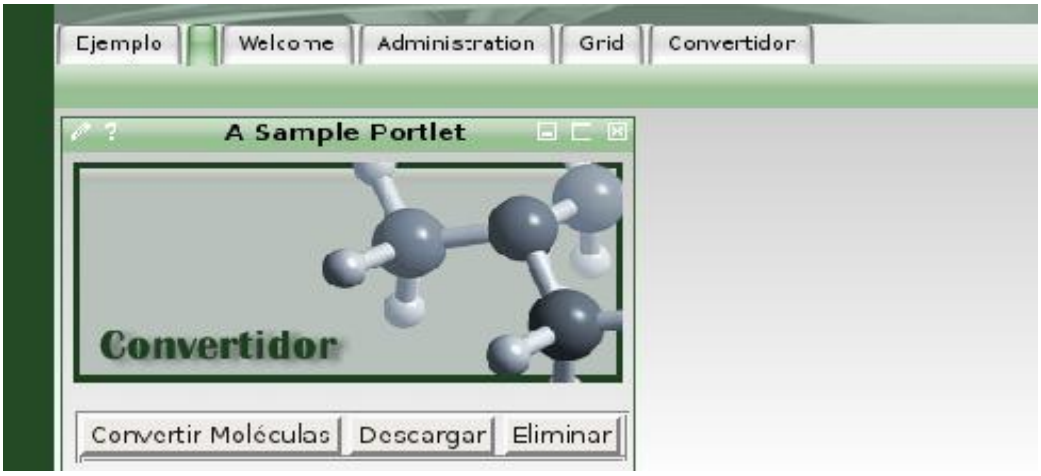


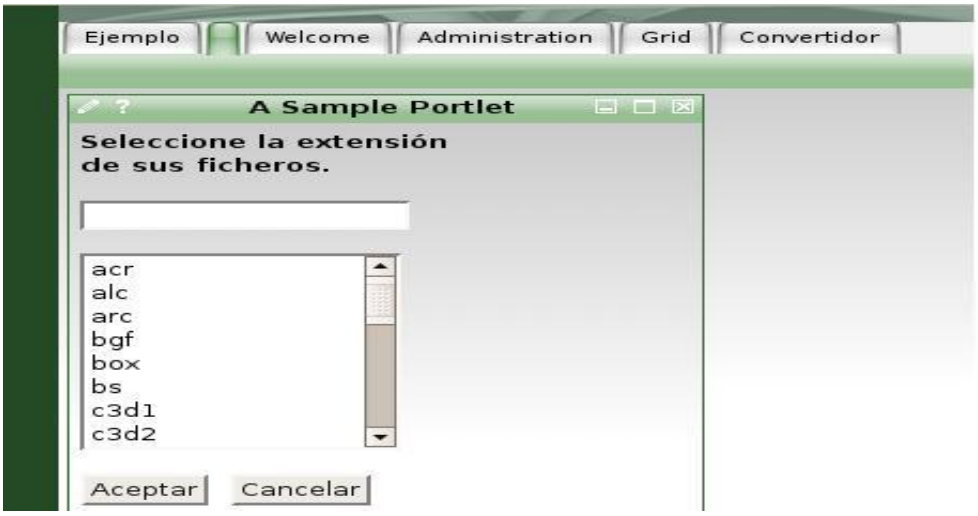
4.2.2 Prototipos de Interfaz de Usuario.

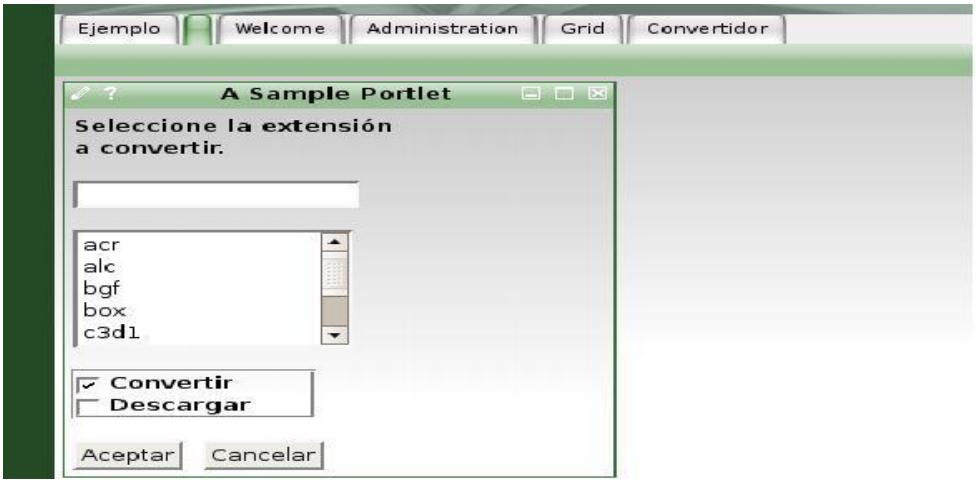


Nosotros realizamos un servicio, el mismo se nombra Convertidor, el cual permite convertir varias moléculas de un tipo de formato a otro. Utiliza la tecnología de portlets JSR 168 y la conversión se realiza utilizando como herramienta el Open Babel.

La interfaces de dicho servicio son las siguientes**:**







(Ver Anexo1)

Para lograr este servicio el método principal es el de convertir molécula (convertirServlet) el cual se explica en el sección **4.3.**

4.3 Código Fuente de las principales clases

General

Este método es el encargado de convertir las moléculas que están almacenadas en el servidor y debe dar como resultado un archivo compactado con las moléculas convertidas, en caso de alguna dificultad con el proceso se le envía un mensaje al usuario, debe además eliminar la carpeta donde están almacenadas las moléculas a convertir.

```
protected void processRequest (HttpServletRequest Request, HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType ("text/html; charset=UTF-8");

 PrintWriter out = response.getWriter ();
       try {

   //-----------------------------------------------------------1
           BabelServiceLocator.ip = "10.34.19.3";
               BabelServiceLocator.port = "3389";
               Babel Proxy client = new Babel Proxy ();

  //-----------------------------------------------------------2
            File folder = new File ("moleculasAconvertir");
            File [] moléculas = folder.listFiles();

           String [] molécula= new String[moleculas.length];
            String [] ficheros Finales= new String[moleculas.length];

  //-----------------------------------------------------------3
            Objects os= request.getSession().getAttribute("extensionInicial");



            String extConvertir = (String) os;
            Objects ol= request.getSession ().getAttribute("extensionAconvertir");
            String extConvertida = (String) ol;

  //-----------------------------------------------------------4
            for (int i = 0; i < moleculas.length; i++)
            {
```

```
            String dirConvertir=(moléculas[i].getAbsolutePath());

             File f = new File(dirConvertir);


          molécula[i]= read(f);
        ficheros Finales[i]=f.getName().substring(0,
f.getName().indexOf("."+extConvertir))+"."+extConvertida;
            }
//-----------------------------------------------------------------5
            String[] conversion= new String[moleculas.length];
            conversión =client.babelConverter(molécula, extConvertir, extConvertida);
            String no convertidos="";
            int cont.=0;
  //-----------------------------------------------------------------6
          if(!new File("moleculasConvertidas").exists())
               new File("moleculasConvertidas").mkdir();
//-----------------------------------------------------------------7
            for(int i=0;i<moleculas.length++)
            {
              if(conversion[i].length()!=0)
              {

             File f2=new File ("moleculasConvertidas/"+ficheros Finales[i]);
              FileWriter wr = new FileWriter (f2);
              wr.write(conversion[i]);
                 wr.flush ();
                 wr.close ();
              }
              else
```

```
                {
                 String temp=ficheros
Finales[i].substring(0,ficherosFinales[i].indexOf("."+extConvertida))+"."+extConvertir;
                 no convertidos+=temp+"\r\n";
                 cont++;
               }
             }
//-------------------------------------------------------------------8
              EliminarDirectorio.deleteDirectory(folder);

//-------------------------------------------------------------------9
              File folder2 = new File("moleculasConvertidas");
              File[] lis = folder2.listFiles();
//-------------------------------------------------------------------10
             if(cont.!=0 && lis. Length==0)
             {
               out.println("No se pudieron convertir los siguientes archivos:\r\n"+ no convertidos);
               EliminarDirectorio.deleteDirectory (new File("moleculasConvertidas"));


             }
//-------------------------------------------------------------------11
             else if(cont.!=0 && lis. Length>0)
             {
     //-----------------------------------------------------------11a


                 out.println("No se pudieron convertir los siguientes archivos:\r\n"+ no convertidos);
                 Objects o_li= request.getSession().getAttribute("fichero Entregar");
                 String nombreFicherofinal = (String) o_li;
```

```
        //-------------------------------------------------------11b
              if(!new File("moleculasConvertidasF").exists())
                new File("moleculasConvertidasF").mkdir();
            //-------------------------------------------------------11c
              FileOutputStream dest = new FileOutputStream ( "moleculasConvertidasF/"
+nombreFicherofinal );

              ZipOutputStream salida = new ZipOutputStream (new BufferedOutputStream (dest));
              byte [] data = new byte[1024];

              for(int i=0;i<lis.length;i++)
              {
               if(lis[i].isFile())
               {

               String filename = lis[i].getPath();
               File fff = new File (filename);
               FileInputStream fi = new FileInputStream(fff );
               BufferedInputStream origin = new BufferedInputStream( fi, 1024);
               ZipEntry entry = new ZipEntry (fff.getAbsoluteFile().getName());
               salida.putNextEntry (entry);

               int count;
               while ((count = origin.read(data, 0, 1024) ) != -1 )
               {
                 salida.write(data, 0, count);


              }
```

```
                origin.close ();
               }
              }

              salida.close ();
        //-------------------------------------------------------11d
              EliminarDirectorio.deleteDirectory (folder2);

            }
 //-------------------------------------------------------------------12
            else
            {
                Object o_li= request.getSession().getAttribute("fichero Entregar");
                String nombreFicherofinal = (String) o_li;
                // Compactando los archivitos

                if(!new File("moleculasConvertidasF").exists())
                   new File("moleculasConvertidasF").mkdir();

                FileOutputStream dest = new FileOutputStream( "moleculasConvertidasF/"
+nombreFicherofinal );


             ZipOutputStream salida = new ZipOutputStream (new BufferedOutputStream (dest));
                byte[] data = new byte[1024];

                for(int i=0;i<lis.length;i++)
                {
```

```
                if(lis[i].isFile())
                {
                 String filename = lis[i].getPath ();
                 File fff = new File (filename);
                 FileInputStream fi = new FileInputStream (fff );
                 BufferedInputStream origin = new BufferedInputStream ( fi, 1024);
                 ZipEntry entry = new ZipEntry (fff.getAbsoluteFile().getName());
                 salida.putNextEntry (entry);

                 int count;
                 while ((count = origin.read (data, 0, 1024)) != -1 )
                 {
                   salida.write (data, 0, count);
                 }

                    origin.close();
                }
               }

               salida.close ();
               EliminarDirectorio.deleteDirectory (folder2);

            }


        }
//-----------------------------------------------------------------13
        catch (Exception error) {
```

```
            // set error flag in session:
            //  request.getSession().setAttribute ("error", error);
            //  out.println (error);
           // throw its further to print in error-page:
         out.println ("Lo sentimos, no hay conexión con el servidor del Babel");
       }
}
```

Explicación:

1 Se realiza la conexión con el web service (este se conecta con el Open Babel para realizar la conversión), de no realizarse la conexión lanza una excepción.

2 Se crean dos arreglos de String, donde se almacenarían, en el primero, los datos de los ficheros a convertir, y en el segundo los de los convertidos, la longitud de los arreglos lo determina la cantidad de ficheros almacenados en la carpeta moléculas a convertir.

3 Se obtienen la extensiones tanto de los ficheros como la que el usuario desea convertir.

4 Se crean los nombres de los posibles archivos convertidos.

5 Se convierte

6 Se crea la carpeta donde se van a almacenar los ficheros convertidos (moleculasConvertidas)

7 Se almacenan los ficheros convertidos en la carpeta moleculasConvertidas, para ello se recorre en un ciclo el arreglo de los ficheros devuelto por el web service.

8 Se elimina la carpeta moleculasAconvertir.

9 Se crea un arreglo de ficheros donde se listan los ficheros que verdaderamente se lograron convertir.

10 Si no se convirtió ninguno, se elimina la carpeta moleculasConvertidas y se le envía un mensaje al usuario para que tenga conocimiento de ello.

11

a) Si no se convirtieron algunos archivos, se le envía un mensaje al usuario informándole de cuales son estos.

b) Se crea la carpeta moleculasConvertidasF, la cual es donde se va a guardar el archivo compactado con todos los ficheros de moléculas convertidas.

c) Se compactan los archivos y el nombre del compactado es el mismo nombre del .Zip que entro el usuario inicialmente.

d) Se elimina la carpeta moleculasConvertidas.

12 Muy parecido al código del punto 11, pero en este caso todas las moléculas son convertidas y no es necesario un mensaje de aviso para el usuario.

13 Se captura la excepción y se envía un mensaje.

4.4 Guía de como desplegar portlets. War en el portal de servicios.

En el Portal se incorporan servicios los cuales son implementados mediante portlets, de ahí la necesidad de explicar bien una guía para desplegar un portlets.

Esta guía describe como desplegar el archivo. War de un portlet compatible con JSR-168 en el portal de servicios. Para ello nos apoyaremos en un ejemplo (Convertidor.war).

Nota: Como requerimientos el usuario debe tener privilegios de administración y el nombre de este archivo debe tener veinte caracteres o menos.

El Gridsphere permite por defecto desplegar portlets, para ello solo tiene que copiar el .war en el servidor Tomcat en el directorio webapps, automáticamente se genera una carpeta con el mismo nombre del .war. Luego en el portal en Administración>Portlet>Portlet Application Manager >Deploy de un portlet wapp escribir el nombre del .war y click en Deploy. Hasta aquí, si el archivo .war cuenta con todos los XML que se requieren se despliega el portlet sin ningún problema. La dificultad radica en que generalmente según la herramienta de desarrollo que se utilice no se genera todos los XML necesarios e incluso los que se generan no siempre cuentan con toda la información precisa y hay que agregarles datos. A continuación mostraremos cuales son los XML que se deben incrementar y los que se deben modificar dentro del archivo .war. **(Ver Anexo 2)**.

Conclusiones

En este capítulo se logra el resultado del diseño, implementando el sistema en términos de componentes, realizando en sí el diagrama de componentes. También se presenta el código fuente de las principales clases y pantallas de la aplicación como por ejemplo prototipos de interfaz de usuario y el mapa de navegación. Por ultimo se creo una guía para adicionar portlets a la plataforma.

Conclusiones Generales

- Se logró la implementación de un Portal Web que satisface las necesidades del Grupo de Bioinformática de la UCI. Está provisto de un ambiente cómodo, fácil de entender, ampliamente definido permitiendo la incorporación de servicios.
- Se definió una guía para desplegar nuevos portlets ya que los servicios que presenta nuestro Portal son implementados mediante portlets.
- Se logró implantar un Servicio Bioinformático que permite convertir un formato de un grupo de moléculas a otro: Convertidor.

Recomendaciones

De manera general, los objetivos trazados al inicio de este trabajo han sido logrados, pero al mismo tiempo, a lo largo del proceso de desarrollo, ha quedado claro que la propuesta es sólo la primera fase de un proyecto que puede ser mucho más ambicioso. Por tanto se declaran las siguientes recomendaciones:

- Mantener el sitio actualizado, incorporándole nuevas funcionalidades.
- Trabajar en nuevas mejoras para el diseño del sitio, teniendo en cuenta la opinión de los usuarios.
- Incorporarle al sitio nuevos portlets.

Referencias Bibliográficas

[1.] [En línea] 2007. [Citado el: 11 de diciembre de 2007.] http://www.solociencia.com/biologia/bioinformatica-concepto.htm

[2.] [En línea] 2007. [Citado el: 11 de diciembre de 2007.] http://es.wikipedia.org/wiki/Servicio

[3.] [Citado el: 2 de febrero de 2008.]http://scholar.google.com.cu/scholar?hl=es&lr=&q=%E2%80%9Cincluyen+tanto+los+servicios+profesionales+vinculados+a+la+instalaci%C3%B3n%2C+mantenimiento%2C+desarrollo%2C+integraci%C3%B3n%2C+etc.+de+software%2C+como+los+de+soporte+t%C3%A9cnico+de+hardware.%E2%80%9D+&btnG=Buscar&lr=

[4.] The BioinfoGRID Project; Revisado Enero 2008. Available from: http://www.

[5.] D. Abramson, R. Buyya, J. Giddy. A Computational Economy for Grid Computing and its Implementation in the Nimrod-G Resource Broker. Future Generation Computer Systems. 18(8), 2002.

[6.] *http://www.wikilearning.com/tutorial/servidores_de_portales_usabilidad_empaquetada-que_es_un_portlet/4084-5*

[7.] Abdelnur, A., & Hepper, S. (2003). *JSR 168—Java Portlet Specification Version 1.0.* Retrieved from http://jcp.org/aboutJava/communityprocess/final/jsr168/index.html.

[8.] Tesis Integración de la plataforma Tarenal con el middleware Globus Toolkit 4.

[9.] GridSphere Web Site. www.gridsphere.org.

[10.] http://www.proactiva-calidad.com/java/patrones/mvc.html. [Citado el: 3 de marzo de 2008.]

[11.] http://jorgesaavedra.wordpress.com/category/patrones-grasp/. [Citado el: 3 de marzo de 2008.]

[12.] Ingeniería de Software 2. "*Patrones de diseño."* UCI. curso 2006_2007. Conferencia.

[13.] Jacobson, I, Booch, G y Rumbaugh, J. *"*El Proceso Unificado de Desarrollo de Software". La Habana : Félix Varela, 2004. Vol. 1.

[14.] Ingeniería de Software 1. "Flujo de Trabajo de Requerimientos." UCI. curso 2007_2008. Conferencia #4.

Bibliografía

1. (s.f.). Recuperado el 5 de febrero de 2008, de http://www.vcell.org/.

2. *"Introducción a Portales Grid"*. (s.f.). Recuperado el 13 de diciembre de 2007, de http://gridcafe.web.cern.ch/gridcafe/gridprojects/portals.html

3. *"Visual Paradigm for UML".* (5 de julio de 2005). Recuperado el 2 de febrero de 2008, de http://www.programacion.com/noticia/1363/

4. Adomian G, A. G. (1984). *Biological System Interactions.Physiological Sciences* (Vols. 81:2838-2940).

5. Alvarez, M. A. (s.f.). *Desarrollo Web "Que es HTML".* Recuperado el 19 de noviembre de 2007, de http://www.desarrolloweb.com/articulos/534.php

6. C.Fellenstein, J. M. (s.f.). *Evolution of grid computing architecture and grid adoption models.IBM Systems Journal,43(4):645,2004.* Recuperado el 19 de noviembre de 2007, de http://www.research.ibm.com/journal/sj/434/josepaut.html.

7. C.Mackenzie, K. R. (s.f.). *Reference model for service oriented architecture 1.0.Technical report,OASIS Standards,2006.* Recuperado el 9 de diciembre de 2007, de http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf.

8. D.Talia. (s.f.). *The open grid services architecture:Where the grid meets the web.IEEE Internet Computing,06(6):67-71,2002.* Recuperado el 10 de diciembre de 2007, de http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1067739.

9. Gracia, J. (s.f.). *"Introducción a PstgreMySQL".* Recuperado el 13 de diciembre de 2007, de http://www.sobl.org/traducciones/practical-postgres/node12.html

10. Jacobo, T. B. (s.f.). *Universidad de Coruña.Diseño e implementación de un portal basado en tecnologías Grid para el acceso a recuersos de supercomputación.* Recuperado el 16 de diciembre de 2007, de http://www.google.com.cu/search?hl=es&q=gridsphere%2Bsecurity&btnG=Buscar+con+Google&meta=lr%3Dlang_es

11. Luscombe NM, G. D. (2001). *What is bioinformatics? An introduction and overview.Yearbook of Medical Informatics. .*

12. M.Roehrig, M. (s.f.). Obtenido de http://www.ogf.org/documents/GFD.11.pdf.

13. Martin Sanchez Fernando, V. R. (s.f.). Recuperado el 12 de diciembre de 2007, de http://www.csi.map.es/csi/tecnimap/tecnimap_2004/comunicaciones/tema_06/6_014.pdf

14. *MSF."Introducción a la Ingienería de Software".* (s.f.). Recuperado el 6 de enero de 2008, de http://rrivera334.blogspot.es/1179459060/

15. *OpenUp/Basic*. (s.f.). Obtenido de http://epf.eclipse.org/wikis/openupsp/.

16. Orallo, H. E. (s.f.). *"El Lenguaje Unificado de Modelado (UML).".* Recuperado el 6 de enero de 2008, de http://www.disca.upv.es/enheror/pdf/ActaUML.PDF

17. R.Yahyapour, P. (s.f.). Recuperado el 2 de febrero de 2008, de http://www.ogf.org/documents/GFD.64.pdf

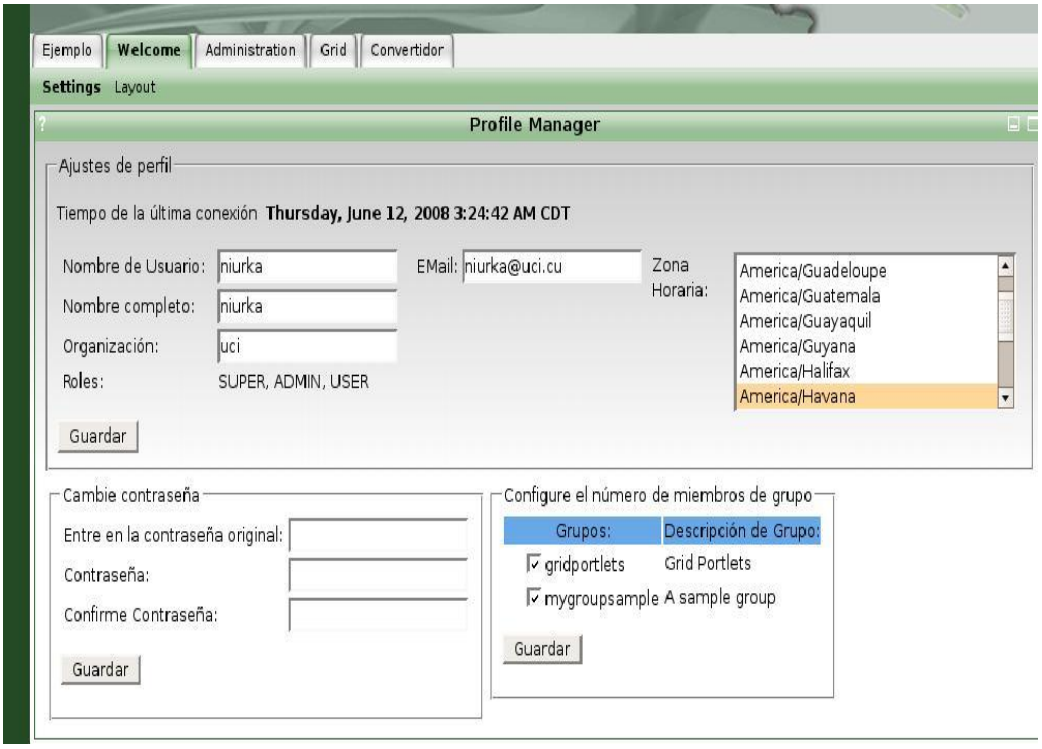
18. Pressman, Roger S. / Madrid, McGraw-Hill. Ingeniería del Software: un enfoque práctico. Parte I y II [En línea] 2002 [Citado el: 11 de Marzo de 2008.] Disponible en: http://bibliodoc.uci.cu/pdf/reg02689.pdf

19. Gracia Joaquín IngenieroSoftware [En línea] Mayo de 2005 [Citado el: 9 de Abril de 2008.] Disponible en:http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php

20. El Arte de Modelar [En línea] [Citado el: 12 de Febrero de 2008.] Disponible en: http://dc.exa.unrc.edu.ar/nuevodc/mbaterias/sistemas/2007/TEORICOS/TEORIA_2_UML_Intro_DC_2007.pdf

Anexos

(Anexo 1)



(Anexo 2)

XML a modificar.

Por defecto el .war generalmente genera estos dos XML:

- web.xml
- portlet.xml

Apoyándonos en el ejemplo veamos cuales son los datos a modificar

portlet.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<portlet-app xmlns="http://java.sun.com/xml/ns/portlet/portlet-app_1_0.xsd"

             version="1.0"

             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

             xsi:schemaLocation="http://java.sun.com/xml/ns/portlet/portlet-app_1_0.xsd">

   <portlet>

      <!-- place portlet description here -->

      <description xml:lang="en">Convertidor2</description>

      <description xml:lang="de">Convertidor2</description>

      <!-- place unique portlet name here -->
```

```
<portlet-name>Convertidor2</portlet-name>

<display-name xml:lang="en">Convertidor2</display-name>

<display-name xml:lang="de">Convertidor2</display-name>

<!-- place your portlet class name here -->



<portlet-class>com.test.Convertidor2</portlet-class>

<expiration-cache>60</expiration-cache>

<!-- place supported modes here -->

<supports>

                    <mime-type>text/html</mime-type>

                    <portlet-mode>VIEW</portlet-mode>

                    <portlet-mode>EDIT</portlet-mode>

                    <portlet-mode>HELP</portlet-mode>

            </supports>

<supports>

   <mime-type>text/wml</mime-type>
```

```
        <portlet-mode>edit</portlet-mode>
        <portlet-mode>help</portlet-mode>
    </supports>
    <supported-locale>en</supported-locale>
    <portlet-info>
        <title>A Sample Portlet</title>
        <short-title>Sample</short-title>
        <keywords>sample</keywords>
    </portlet-info>
    <!-- place portlet preferences here -->
    <portlet-preferences>
        <preference>
            <name>myPref</name>
            <value>avalue</value>
            <read-only>true</read-only>
        </preference>
```

```
    </portlet-preferences>
</portlet>
<custom-portlet-mode>
    <description xml:lang="en">Pre-defined custom portlet mode CONFIGURE</description>
    <portlet-mode>CONFIGURE</portlet-mode>
</custom-portlet-mode>
<user-attribute>
    <description xml:lang="en">User Name</description>
    <name>user.name</name>
</user-attribute>
<user-attribute>
    <description xml:lang="en">User Id</description>
    <name>user.id</name>
</user-attribute>
<user-attribute>
    <description xml:lang="en">User Full Name</description>
```

```
    <name>user.name.full</name>
</user-attribute>
<user-attribute>
    <description xml:lang="en">User E-Mail</description>
    <name>user.email</name>
</user-attribute>
<user-attribute>
    <description xml:lang="en">Company Organization</description>
    <name>user.organization</name>
</user-attribute>
<user-attribute>
    <description xml:lang="en">Last Login Time</description>
    <name>user.lastlogintime</name>
</user-attribute>
<user-attribute>
    <description xml:lang="en">Timezone</description>
```

```
    <name>user.timezone</name>
</user-attribute>
<user-attribute>
    <description xml:lang="en">Preferred Locale</description>
    <name>user.locale</name>
</user-attribute>
<user-attribute>
    <description xml:lang="en">Preferred Theme</description>
    <name>user.theme</name>
</user-attribute>
<!--
    <security-constraint>
        <portlet-collection>
            <portlet-name>TimeZoneClock</portlet-name>
        </portlet-collection>
        <user-data-constraint>
```

```
            <transport-guarantee>CONFIDENTIAL</transport-guarantee>

        </user-data-constraint>

    </security-constraint>

  -->

</portlet-app>
```

Este archivo debe tener esta estructura, verificar según su portlet y agregar los datos que pudieran faltar, por ejemplo asegurarse que en la etiqueta <portlet-info> este <keywords>sample</keywords> así como que estén presentes las etiquetas <portlet-preferences>,<user-attribute>
 <custom-portlet-mode>, con todo lo que se expone en ellas se puede tomar como referencia el ejemplo.

web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>



<!DOCTYPE web-app

   PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"

   "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>

   <display-name>Convertidor</display-name>
```

```
<description>
    Provides Convertidor
</description>

<context-param>
    <!-- preferred repository selector. "preferred" because
    if one is already installed, this choice is ignored. -->
    <param-name>log4j-selector</param-name>
    <param-value>org.apache.log4j.selector.ContextJNDISelector</param-value>
</context-param>
<context-param>
    <!-- relative path to config file within current webapp -->
    <param-name>log4j-config</param-name>
    <param-value>WEB-INF/classes/log4j.properties</param-value>
</context-param>
```

```
<!-- uncomment only if using in non-GridSphere container
<listener>
    <listener-class>org.gridlab.gridsphere.provider.portlet.jsr.PortletServlet</listener-class>
</listener>
 -->

<listener>
    <listener-class>org.apache.log4j.servlet.InitContextListener</listener-class>
</listener>

<servlet>
    <servlet-name>PortletServlet</servlet-name>
    <servlet-class>org.gridlab.gridsphere.provider.portlet.jsr.PortletServlet</servlet-class>
</servlet>
```

```
<servlet-mapping>
   <servlet-name>PortletServlet</servlet-name>
   <url-pattern>/jsr/convertidor</url-pattern>
</servlet-mapping>
<servlet>
    <servlet-name>Fichero</servlet-name>
    <servlet-class>Servlets.Fichero</servlet-class>
</servlet>
<servlet>
    <servlet-name>ficheroSerlevt</servlet-name>
    <servlet-class>Servlets.ficheroServlet</servlet-class>
</servlet>
<servlet>
    <servlet-name>convertirSerlevt</servlet-name>
    <servlet-class>Servlets.convertirServlet</servlet-class>
</servlet>
```

```
<servlet>
    <servlet-name>descargarSerlevt</servlet-name>
    <servlet-class>Servlets.descargarServlet</servlet-class>
</servlet>
<servlet>
    <servlet-name>sesionesServlet</servlet-name>
    <servlet-class>Servlets.sesionesServlet</servlet-class>
</servlet>
<servlet>
    <servlet-name>extension_Convertir</servlet-name>
    <servlet-class>Servlets.extensionServlet</servlet-class>
</servlet>
<servlet>
    <servlet-name>extensionServlet</servlet-name>
    <servlet-class>Servlets.extensionServlet</servlet-class>
</servlet>
```

```
<servlet>
    <servlet-name>verificar_carpetaServlet</servlet-name>
    <servlet-class>Servlets.verificar_carpetaServlet</servlet-class>
</servlet>
<servlet>
    <servlet-name>extension_0Servlet</servlet-name>
    <servlet-class>Servlets.extension_0Servlet</servlet-class>
</servlet>
<servlet>
    <servlet-name>eliminarServlet</servlet-name>
    <servlet-class>Servlets.eliminarServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>Fichero</servlet-name>
    <url-pattern>/Fichero</url-pattern>
</servlet-mapping>
```

```
<servlet-mapping>
    <servlet-name>ficheroSerlevt</servlet-name>
    <url-pattern>/ficheroSerlevt</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>convertirSerlevt</servlet-name>
    <url-pattern>/convertirSerlevt</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>descargarSerlevt</servlet-name>
    <url-pattern>/descargarSerlevt</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>sesionesServlet</servlet-name>
    <url-pattern>/sesionesServlet</url-pattern>
</servlet-mapping>
```

```
<servlet-mapping>
    <servlet-name>extension_Convertir</servlet-name>
    <url-pattern>/extension_Convertir</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>extensionServlet</servlet-name>
    <url-pattern>/extensionServlet</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>verificar_carpetaServlet</servlet-name>
    <url-pattern>/verificar_carpetaServlet</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>extension_0Servlet</servlet-name>
    <url-pattern>/extension_0Servlet</url-pattern>
</servlet-mapping>
```

```
<servlet-mapping>
    <servlet-name>eliminarServlet</servlet-name>
    <url-pattern>/eliminarServlet</url-pattern>
</servlet-mapping>

<mime-mapping>
  <extension>wbmp</extension>
  <mime-type>image/vnd.wap.wbmp</mime-type>
</mime-mapping>

<mime-mapping>
  <extension>wml</extension>
  <mime-type>text/vnd.wap.wml</mime-type>
</mime-mapping>

<mime-mapping>
```

```
    <extension>wmls</extension>
    <mime-type>text/vnd.wap.wmlscript</mime-type>
  </mime-mapping>

  <env-entry>
     <description>JNDI logging context for this app</description>
     <env-entry-name>log4j/logging-context</env-entry-name>
     <env-entry-value>convertidor</env-entry-value>
     <env-entry-type>java.lang.String</env-entry-type>
  </env-entry>

</web-app>
```

Este archivo también debe tener esta estructura, las etiquetas que falten agregarlas y ajustarlas según su portlet desarrollado.

- **XML a agregar.**

Los XML que se tienen que agregar son:

-PortletServices.xml

-layout.xml

-group.xml

-Gridsphere-portlet

PortletServices.xml

```
<!--


    Portlet Services XML Descriptor



    Edit this file to add/modify Portlet Services



    $Id: PortletServices.xml.tpl 4496 2006-02-08 20:27:04Z wehrens $

-->



<portlet-services>



    <service>
```

```
        <name>Example Service</name>

        <description>Provides Capabilities</description>

        <interface>com.mycom.ExampleService</interface>

        <implementation>com.mycom.impl.ExampleServiceImpl</implementation>

    </service>



    <service>

        <name>Secure Example Service</name>

        <user-required>true</user-required>

        <description>Provides Secure Capabilities</description>

        <interface>com.mycom.SecureService</interface>

        <implementation>com.mycom.SecureServiceImpl</implementation>

    </service>



</portlet-services>
```

layout.xml

```
<portlet-tabbed-pane>

   <portlet-tab label="Convertidor">

         <title lang="en">Convertidor</title>

      <portlet-tabbed-pane style="sub-menu">

         <portlet-tab label="zoneclocktab">

               <title lang="en">Convertidor</title>

               <title lang="de">Convertidor</title>

            <table-layout>

               <row-layout>

                  <column-layout>

                     <portlet-frame label="convertidor">

                        <portlet-class>com.test.Convertidor2</portlet-class>

                     </portlet-frame>

                  </column-layout>

               </row-layout>
```

```
            </table-layout>

        </portlet-tab>

    </portlet-tabbed-pane>

  </portlet-tab>

</portlet-tabbed-pane>
```

group.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<!--

 Sample group layout



 $Id: group.sample.xml.tpl 4496 2006-02-08 20:27:04Z wehrens $

-->

<portlet-group>

  <group-name>mygroupsample</group-name>

  <group-description>A sample group</group-description>
```

```
    <group-visibility>PUBLIC</group-visibility>

    <portlet-role-info>

        <portlet-class>com.test.Convertidor2</portlet-class>

        <required-role>USER</required-role>

    </portlet-role-info>

    <portlet-role-info>

        <portlet-class>com.test.Convertidor2</portlet-class>

        <required-role>ADMIN</required-role>

    </portlet-role-info>

</portlet-group>
```

Gridsphere-portlet.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<portlet-app-collection>



<portlet-app-def>
```

```
<portlet-app id="org.gridlab.gridsphere.provider.portlet.jsr.PortletServlet">
    <portlet-name>JSR Portlet Servlet</portlet-name>
    <servlet-name>PortletServlet</servlet-name>
</portlet-app>



<concrete-portlet-app id="org.gridlab.gridsphere.provider.portlet.jsr.PortletServlet.1">
    <concrete-portlet>
        <portlet-name>Portlet Servlet</portlet-name>
        <default-locale>en</default-locale>
        <language locale="en">
            <title>Portlet Servlet</title>
            <title-short>Portlet Servlet</title-short>
            <description>A JSR Portlet Loader</description>
            <keywords>portlet servlet</keywords>
        </language>
    </concrete-portlet>
```

```
</concrete-portlet-app>



</portlet-app-def>



</portlet-app-collection>
```

Los XML gridsphere-portlet.xml y PortletServices.xml se agregan tal y como están presentes en el ejemplo, sin ninguna modificación. Mientras que los XML restantes si se modifican de acuerdo a las características de como desee customizar su portlet.

Además de estas orientaciones debe agregar en WEB-INF/lib la librería siguiente gridsphere-ui-tags-2.2.jar, esta librería se encuentra a disposición en nuestro portal, así como también en WEB-INF clases agregar la propiedad log4j.properties con el siguiente contenido:

```
# Set root category priority to ERROR and its only appender to A1.

log4j.rootCategory=ERROR, A1



# A1 is set to be a ConsoleAppender.

log4j.appender.A1=org.apache.log4j.ConsoleAppender
```

```
# A1 uses PatternLayout.

log4j.appender.A1.layout=org.apache.log4j.PatternLayout

log4j.appender.A1.layout.ConversionPattern=%-4r [%t] (%F:%L) %-5p %c %x - %m%n



# Add packages to log

#log4j.logger.org.myorg=DEBUG
```

Una ves que su archivo .war este modificado con todos estos datos realizar los pasos que se indican en el inicio de la guía.

Glosario

E

EJB: *Enterprise Java Beans. Son pedacitos de código, clases chiquiticas, que están dentro de la EJC (Enterprise Java Chauchas) formando arrays de porotos. No se debe abusar de la implementación de esta tecnología en un proyecto ya que el exceso de "beans"produce "fart errors" que son desbordamientos de metano en el buffer digestivo del compilador.*

F

Framework: Un framework se puede considerar como una aplicación genérica incompleta y configurable a la que podemos añadirle las últimas piezas para construir una aplicación concreta. Los objetivos principales que persigue un framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones. Un framework Web, por tanto, podemos definirlo como un conjunto de componentes (por ejemplo clases en java y descriptores y archivos de configuración en XML) que componen un diseño reutilizable que facilita y agiliza el desarrollo de sistemas Web.

Filogenia: Es la parte de la biología que estudia la evolución de las especies de forma global, en contraposición a la ontogenia, que estudia la evolución del individuo.

G

GRASP: Es un acrónimo que significa General Responsibility Asignment Software Patterns (Patrones generales de software para asignar responsabilidades).

H

HTML: *(*Hypertext Markup Language): Lenguaje basado en marcas que indican las características del texto, utilizado para definir documentos de hipertexto en webs.

HTTP: Es el conjunto de reglas para intercambiar archivos (texto, gráfica, imágenes, sonido, video y otros archivos multimedia) en la World Wide Web.

J

Java script: Es un lenguaje interpretado orientado a las páginas web, para realizar tareas y operaciones en el marco de la aplicación cliente.

O

OpenUP (Open Unified Process en inglés o Proceso Unificado Abierto)*:* Constituye una metodología ágil. Este proceso puede ser desarrollado para hacer frente a una extensa diversidad de tipos de proyectos. Además OpenUP sostiene las mismas características de RUP.

P

PHP: Lenguaje interpretado usado para la creación de aplicaciones para servidores, o creación de contenido dinámico para sitios web.

Portal WEB: Es un sitio web cuyo objetivo es ofrecer al usuario, de forma fácil e integrada, el acceso a una serie de recursos y de servicios. Están dirigidos a resolver necesidades específicas de un grupo de personas o de acceso a la información y servicios de a una institución pública o privada.

PHP: *(*PHP Hypertext Preprocessor): Lenguaje de programación para el desarrollo de webs dinámicas, con sintaxis parecida a la que originalmente se conocía como Personal Hombre Page tools, herramientas para páginas personales (en Internet).

POJO: (acrónimo de Plain Old Java Object) utilizada por programadores lenguaje de programación Java para enfatizar el uso de clases simples y que no dependen de un framework en especial.

R

Racional Unified Process (RUP): Metodologías utilizada para el desarrollo de software. Actualmente es una de las más utilizadas a nivel mundial.

X

XML: *(*Extensible Markup Language): Lenguaje de marcas extensible, es un metalenguaje extensible de etiquetas desarrollado por el Word Wide Web Consortium.