Universidad de las Ciencias Informáticas

Facultad 6

Título: "Portlet para la gestión de un clúster Beowulf"

Trabajo de Diploma para optar por el título de

Ingeniero Informático

Autor:

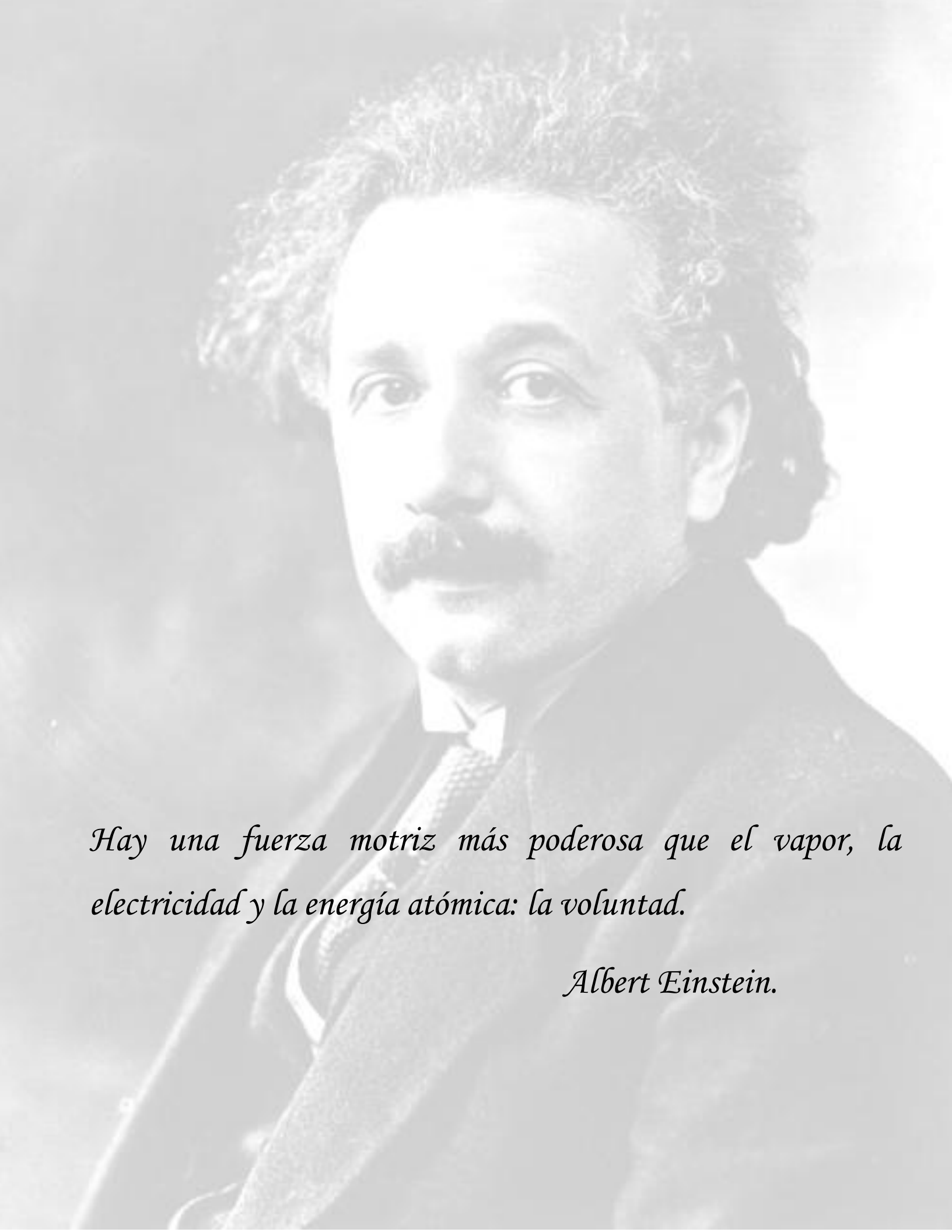
Ernesto Antonio Rodríguez Rodríguez

Tutores:

Ing. Rigoberto L. Salgado Reyes

Ing. Cesar Raúl García Jacas

La Habana, mayo de 2013
"Año 55 de la Revolución"



Hay una fuerza motriz más poderosa que el vapor, la electricidad y la energía atómica: la voluntad.

Albert Einstein.

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Ing. Rigoberto L. Salgado Reyes

_____

Firma del Tutor

Ing. Cesar Raúl García Jacas

_____

Firma del Tutor

Ernesto Antonio Rodríguez Rodríguez

_____

Firma del Autor

DATOS DE CONTACTO

TUTORES:

Ing. Rigoberto L. Salgado Reyes

Universidad de las Ciencias Informáticas, La Habana, Cuba

Email: rlsalgado@uci.cu

Ing. Cesar Raúl García Jacas

Universidad de las Ciencias Informáticas, La Habana, Cuba

Email: crjacas@uci.cu

AGRADECIMIENTO

Gracias a todas aquellas personas que de una forma u otra me han apoyado siempre, me han guiado por un buen camino y aconsejado de forma correcta, pues sin ellos no sería quien soy. Mencionarlos a todos creo que sería imposible pues son muchos los que de una forma u otra han contribuido a la causa de hacer un hombre de bien con mi persona.

Agradecer en especial a mis amigos del barrio, que nunca olvidaron quien soy y su amistad sigue tan fresca como siempre a pesar de la distancia y el tiempo fuera de casa.

Gracias también a todas las bellas personas que conocí aquí en la UCI, especialmente a los más allegados a mí que siempre me extendieron la mano en las situaciones difíciles que me presentó la vida.

Nunca voy a olvidar mi primer año, ese año donde todo te parece grande y estrepitoso, mi primer grupo 6106, quienes juntos dimos los primeros pasos como universitarios y donde también estaban mis primeros compañeros con los que compartí hasta segundo año buenos y malos momentos.

A los amigos que fui conociendo y creando mientras transcurrían los años en la UCI a quienes también debo mucho y a los cuales no voy a olvidar jamás.

A mis compañeros de quinto año, que me soportaban cuando preguntaba mucho y a los que yo soportaba cuando preguntaban también les quiero agradecer el permitirme compartir esos momentos juntos, porque yo sé que no olvidarán esos días que seguramente no volverán jamás.

A mi tutor principal Rigo, mis más profundos y verdaderos agradecimientos, que sin su perseverancia y su inteligencia no hubiese sido capaz de realizar este trabajo de diploma, el cual pone fin a mi carrera como estudiante. Gracias por guiarme a los lugares indicados donde encontrar información, por cuando era necesario "asustarme" y ponerme a correr porque hoy me doy cuenta que fue necesario y solo era por mí bien.

A mi novia Lismey quien me ha ayudado a ser una mejor persona, a organizar mis regueros y siempre está ahí cuando más lo necesito, mis más sinceros agradecimientos por ser paciente y comprensiva conmigo.

Quiero hacer un alto especial para hablar de mi familia, familia sin la cual no sé qué sería de mí, no sé si hoy estuviera aquí; a mi tía Marlen, a mis tíos Alexander y Carlos, a mi abuelo Alberto, y a muchos más que por cosas de la vida no estamos todos los días juntos pero también están pendientes de mí, mi más profundo agradecimiento para ellos que son la clave de mi existir.

A Martha, Irhomis y a mi hermano Erier, a esas personas que admiro y quiero mucho por sus todos sus actos de afecto y amor hacia mí.

A mi hermano Erick, mi hermano mayor, que a pesar de no vivir conmigo ha hecho función de padre para mí y me sirve de guía en el intrincado camino de la vida, un hombre de muchas leyes pero de un carácter espectácular.

A mi papá que a pesar de no vivir juntos nunca se ha apartado de mí como yo tampoco de él, un hombre serio, pero risueño al que me parezco mucho, y me encanta escuchar hablar, y hacerle chistes para que se ría cuando tenemos la oportunidad de estar cerca ,porque aunque no compartí mi infancia con él a tiempo completo, sé que le debo la vida tanto a él como a mi mamá y ese es el regalo más grande que me ha dado.

A Eugenio, mi otro papá, a quien tengo que agradecer mi carácter, mi cultura, mi visión de la vida, alguien que sin haberme engendrado, tuvo la osadía de tomar las riendas de mi vida y hacer conmigo lo mismo que hubiese hecho con su hijo, alguien que supo tomar el lugar de mi papá y hacerlo suyo sin pretensión alguna de sustituirlo sino de hacer lo que él también hubiese hecho conmigo.

A mi abuela Ana, mi "Mima", que es la abuela más bella que existe en el mundo entero, que es capaz de dar la vida por mí como yo la doy por ella sin pensarlo ni un segundo, que es mi otra mamá, que siempre está y estará presente en mi corazón y en mi pensamiento. Que me entregó su amor como a su más preciado hijo y que gracias a Dios aún puedo disfrutar de su compañía, y espero que eso dure por siempre.

A mi mamá, mi "Ma", quien ha dedicado la mayor parte de su vida a cuidar y enseñar a su único hijo, que a veces discutimos porque me quiere sobreproteger pero eso no es otra cosa que amor de madre, amor puro y verdadero, del que solo ella puede experimentar, pero así mismo la amo yo, con todas las fuerzas y hasta el final, porque madre hay una sola en el mundo y si tengo la oportunidad de estar a su lado me siento en la obligación de quererla y de cuidarla, porque como ella nadie nunca me querrá, la quiero sin importar lo que pase ni en el lugar que esté porque así me enseñó ella, porque así aprendí de las personas que me rodean, aprendí a amar y a valorar las cosas que hacen los demás por mí, por eso escribí estas palabras de agradecimiento para que todas las personas que comparten mi vida no tengan dudas que para mí también es un placer y una suerte tenerlos a mi lado.

DEDICATORIA

Quiero dedicar este trabajo principalmente a las personas más allegadas: mi mamá, mi papá, a Eugenio, a mi abuela y a todas esas personas que me impulsaron a llegar hasta aquí. A mi tutor, por ayudarme en todo lo que estuvo en sus manos y un poco más allá también. A la Revolución cubana por darme la oportunidad de ser alguien en la vida sin costo alguno, solo el sacrificio y la entrega diaria. A mis colegas de siempre que me han apoyado y ayudado sin ningún tipo de interés. Y muy especial a mi mamá, mi abuela y a "Uge" que siempre están presentes, en las buenas y en las malas, sacrificando lo suyo para que yo llegue a ser lo que siempre ellos soñaron; hombre preparado, culto, inteligente, que sepa afrontar la vida de frente y eso creo que lo lograron.

RESUMEN

Las Tecnologías de la Información y las Comunicaciones (TICs) juegan un papel fundamental en la transformación de la sociedad, especialmente para áreas del conocimiento como la Bioinformática. La siguiente investigación se enmarca en el desarrollo de una aplicación que proporcione una interfaz visual para la gestión de las tareas y usuarios del Clúster Beowulf presente en el Departamento de Bioinformática de la Universidad de las Ciencias Informáticas. Esta aplicación se desarrolla con el fin de proporcionar al usuario facilidades de comunicación con el clúster para así hacer más amena la comunicación investigador-aplicación. Con el fin de desarrollar la aplicación se definieron las tecnologías y programas para desarrollar el sistema. El proceso estuvo guiado por la metodología de desarrollo de software OpenUP y se utilizó como lenguaje de programación Java. Finalmente se obtuvo un portlet que permite a los investigadores gestionar sus tareas corridas sobre el clúster, también posibilita a los administradores, la gestión de los usuarios del clúster. El sistema se integra a la Plataforma de Servicios Bioinformáticos de la Universidad de las Ciencias Informáticas (UCI) y puede ser reutilizado por otras aplicaciones indistintamente de la tecnología en que estas han sido elaboradas.

Palabras Claves: Bioinformática, Clúster Beowulf, Gestión de Tareas, Plataforma de Servicios Bioinformáticos.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

La informática es una ciencia que está en un constante desarrollo, en la actualidad los más importantes procesos de las grandes ciudades se han automatizado y dependen en su gran mayoría de las máquinas. También el estudio de las ciencias y el desarrollo de experimentos se han agilizado y facilitado con las herramientas informáticas y computadoras existentes. Los estudios en la rama de la biología y la medicina son un ejemplo de esto, en tiempos pasados los experimentos biológicos demoraban meses incluso años en arrojar resultados. En nuestros días esto ha cambiado mucho, ya que existen herramientas y tecnologías que pueden por ejemplo simular experimentos que antes se ejecutaban en tiempo real y eran muy costosos en cuanto a recursos y tiempo, en cambio si se cuenta con herramientas capaces de simular estos experimentos el trabajo científico se hace mucho más fácil y factible. Las herramientas de las que hablamos anteriormente suelen ser específicamente software de cálculo y/o búsqueda, este tipo de aplicaciones requiere de una alta disponibilidad de cómputo por lo que es necesario contar con potentes computadores o alguna alternativa a estos para así hacer un eficaz uso de las tecnologías.

En el Centro de Tecnologías de Gestión de Datos (DATEC) de la Universidad de las Ciencias Informáticas (UCI), específicamente el departamento de Bioinformática (BioSoft) se especializa en el desarrollo de software para la investigación científica en la rama de la biología y la medicina. Dicho departamento cuenta con un *Clúster Beowulf*[1] para la corrida de aplicaciones y algoritmos que especifican una alta disponibilidad de cómputo, este clúster se utiliza como alternativa al no contar con grandes computadores que satisfagan las necesidades del departamento. En estos momentos BioSoft está inmerso en el desarrollo de un Portal de Servicios Bioinformáticos el cual será de gran utilidad para las investigaciones en las ramas de la biología y la medicina en nuestro país. Este portal integrará un grupo de aplicaciones las cuales permitirán a los científicos realizar cálculos y simular experimentos de forma controlada y organizada pudiendo mejorar así los resultados de los mismos. Estas aplicaciones se adicionarán al portal en forma de *portlet*[2] lo cual facilita la gestión de las mismas pues permite al usuario visualizar lo que estime conveniente.

[1] Un clúster Beowulf es un grupo de computadoras compatibles con IBM, que utiliza software de código abierto. Para que un grupo de computadoras pueda ser considerado un "Beowulf" deben cumplir ciertos requisitos.

[2] Es un componente Web basado en la tecnología Java que procesa los pedidos del usuario y genera un contenido dinámico final, el contenido generado por un Portlet es llamado fragmento y su ciclo de vida es manejado por el contenedor de Portlets.

El clúster con que cuenta el departamento de Bioinformática sería una de las columnas que sostendrían al Portal de Servicios Bioinformáticos pues muchos de los softwares que prestarán servicio desde el portal estarían desplegados sobre el clúster ya que estos requieren de una cantidad de cómputo razonablemente mayor a la que puede brindar cualquier ordenador personal.

La gestión de las tareas, recursos y usuarios del clúster se realiza actualmente mediante una interfaz a líneas de comando o consola, la cual dificulta la interacción entre el usuario y el clúster ya que el usuario debe tener conocimientos avanzados de informática para poder realizar acciones sobre la interfaz actual. Partiendo del vínculo que existe entre las aplicaciones del Portal de Servicios Bioinformáticos y el clúster, se estima conveniente crear una interfaz gráfica mediante la cual se puedan gestionar las tareas, recursos y usuarios del clúster, y añadir esta interfaz al Portal de Servicios Bioinformáticos. Dicha interfaz permitirá realizar las mismas funciones que se realizan actualmente sobre el clúster y a su vez mejorará el proceso de interacción usuario-aplicación permitiendo que usuarios no tan experimentados en el campo de la informática puedan operar sobre el clúster con mayor facilidad.

Según lo planteado anteriormente surge como **problema a resolver:** La implementación de un *portlet* que permita la integración del clúster al portal de servicios.

El portlet permitirá, con la integración como un servicio más del portal, la gestión de las tareas y usuarios del clúster.

Se plantea como Objeto de Estudio:

- Portlets para la gestión de tareas y recursos.

Enmarcado en el Campo de Acción:

- Portlets para la gestión de tareas y recursos en clústeres Beowulf.

Se define como Objetivo General:

- Desarrollar un portlet que permita la gestión de tareas y recursos de un clúster Beowulf.

Objetivos específicos:

- Diseñar el portlet para la gestión de tareas y recursos del clúster Beowulf.
- Implementar el portlet para la gestión de tareas y recursos del clúster Beowulf.
- Integrar el portlet al portal de servicios de Bioinformática.
- Realizar pruebas de funcionalidad del portlet.

Con las siguientes Tareas a Cumplir:

- Estudio del framework Liferay como contenedor de portlets.
- Diseño del portlet para la gestión de tareas y recursos del clúster.
- Implementación del portlet para la gestión de tareas y recursos del clúster.
- Integración del portlet al portal de servicios.
- Realización de pruebas de funcionalidad del portlet.

Esperando obtener los Posibles Resultados:

- Se obtendrá un portlet que posibilitará la gestión de tareas y recursos de un clúster Beowulf.

El trabajo tiene la estructura siguiente:

CAPITULO 1. Fundamentación Teórica.

Se exhibe el marco teórico estudiado y seleccionado para el presente trabajo. Se presenta un panorama general de los conceptos y herramientas imprescindibles para lograr los objetivos trazados.

CAPITULO 2. Características del Sistema.

Se establece una breve descripción de la solución planteada, el modelo de dominio del sistema, así como los requisitos funcionales y no funcionales. Se describen los diagramas de casos de usos del sistema y se definen los actores del sistema.

CAPITULO 3. Diseño de la Aplicación.

Se describe la representación arquitectónica del sistema. Se detallan los principios de diseño que se tuvieron en cuenta durante el desarrollo de la aplicación. Se hace énfasis en el patrón de arquitectura utilizado, así como los patrones de diseño que fueron empleados. Se realiza el diagrama de clases del diseño dando respuesta a la solución que se propone.

CAPITULO 4. Implementación y Prueba.

Se describe la implementación del sistema y se muestran los diagramas de componentes. Además se muestran los casos de prueba de caja negra para cada caso de uso con el objetivo de validar el correcto funcionamiento de la aplicación.

CAPÍTULO 1.FUNDAMENTACIÓN TEÓRICA.

En este capítulo, se realiza una búsqueda y análisis de información para determinar cuáles son las herramientas y metodologías utilizadas que dan respuesta al problema planteado. Además se realiza una breve descripción de conceptos aplicados que se contemplan en el estado del arte de la investigación desarrollada.

1.1 Gestión de un Clúster Beowulf.

Un sistema de gestión de colas y recursos de computación es una herramienta que permite al usuario aprovechar las capacidades del sistema de computación de forma más eficiente. También que el administrador del sistema pueda asegurar un uso equitativo del sistema cumplimiento de las políticas de uso (1).

1.1.1 Sistemas de gestión de colas.

Un sistema de gestión de colas es el encargado de gestionar la ejecución de los trabajos enviados a un clúster. Este sistema planifica la ejecución de las tareas y proporciona agilidad, organización y control en la ejecución.

¿Qué hace un sistema de gestión de colas?

1. **Poner en cola:**
   Pone en cola las tareas para ser ejecutadas en un clúster o computador. El usuario envía tareas o “trabajos” al gestor de recursos, y éstos son puestos en cola hasta que el sistema esté preparado para ejecutarlos.
2. **Planificar:**
   Planifica para seleccionar correctamente qué trabajos se van a ejecutar, cuándo y dónde, conforme a una determinada política con el objetivo de maximizar los recursos, el tiempo de computación y tiempo de los usuarios.
3. **Monitorizar:**
   Realizar un seguimiento de la utilización de los recursos del sistema y asegurar que se están usando correctamente y aplicando las políticas de uso.

A continuación se brinda una breve descripción de algunos de los gestores de colas más usados:

- **SunGridEngine (SGE)** (2)

SunGridEngine (SGE) es un sistema de encolamiento de procesos distribuido de código abierto desarrollado por Sun Microsystems. Como su propio nombre indica el objetivo de este software es la creación y gestión de una arquitectura de Computación mediante el uso de la red. SGE se usa frecuentemente en clústeres de Computación de alto rendimiento (HPC). Este software también se encarga de la aceptación, programación, envío y gestión de la ejecución remota y distribuida de un gran número de tareas en espacio de usuario ya sean individuales, paralelos o interactivos. Entre las funcionalidades de SGE también se encuentra la posibilidad de gestionar y programar la reserva de recursos distribuidos a lo largo de la plataforma como procesadores, memoria, espacio de disco y licencias de software.

- **Portable Batch System (PBS)** (3)

PBS Professional es desarrollado por el mismo equipo que originalmente diseñó y desarrolló PBS para la NASA. El equipo de ingeniería básica de tecnologías de redes Altair, ha llevado a PBS a los ordenadores de todo el mundo, incluyendo algunas de las supercomputadoras más grandes que existen. PBS permite la gestión de recursos y planificación de tareas así como la optimización en clústeres y supercomputadoras, el paso de mensajes, así como la computación paralela y distribuida de alto rendimiento son los principales aspectos que se tuvieron en cuenta a la hora de diseñar PBS.

- **OpenPBS** (4)

OpenPBS es capaz de ejecutar programas que corran en una sola máquina, o en varias mediante el protocolo MPICH[3]. En caso de que un usuario necesite ejecutar un programa, OpenPBS se ocupa automáticamente de buscar un nodo del clúster que no esté siendo utilizado, y realizar dicha ejecución en el nodo. De igual manera, en caso de querer ejecutar una aplicación distribuida, se buscará el número necesario de nodos para realizar la ejecución. En caso de que actualmente no pueda atenderse la ejecución de un programa por estar todos los nodos ocupados, OpenPBS introduce nuestra petición en una cola en espera de que se liberen los recursos necesarios.

[3] Protocolo mediante el que se comunican los nodos del clúster.

- **Torque** (5)

El gestor de recursos Torque, proporciona control sobre los trabajos por lotes y los recursos de computación distribuida. Es un avanzado producto de código abierto basado en el proyecto original PBS e incorpora lo mejor de la comunidad y el desarrollo profesional. Se incorporan avances significativos en las áreas de escalabilidad, fiabilidad y funcionalidad y se encuentra actualmente en uso en decenas de miles de clústeres o supercomputadores, académicos y sitios comerciales en todo el mundo. Torque, puede ser libremente utilizado, modificado y distribuido bajo las restricciones de la licencia incluida.

A continuación un conjunto de características que ofrece Torque con respecto a otros gestores de recursos en las siguientes áreas:

- **Tolerancia de fallos**
  - ✓ Condiciones adicionales de fracaso comprobado/manipulado.
  - ✓ Chequeo del estado de cada nodo.
- **Interfaz de Planificación**

  El planificador proporciona una Interfaz de consulta ampliada con información adicional y más precisa. Su interfaz permite al programador un mayor control sobre el comportamiento y empleo de los atributos. Permite la recopilación de estadísticas de trabajos terminados.
- **Escalabilidad**

  Cuenta con un servidor significativamente mejorado. Torque tiene capacidad para manejar grandes grupos de nodos y puestos de trabajo. También tiene capacidad para manejar trabajos más grandes que abarcan grandes cantidades de procesadores. Mayor capacidad de respuesta y la fiabilidad de la comunicación multi-threading basada en TCP.
- **Usabilidad**

  Un registro más legible por los usuarios con extensas incorporaciones.

El equipo que se desempeña a cargo de la gestión y mantenimiento del clúster decidió usar Torque como sistema de encolamiento dado que presenta una gama de características y ventajas sobre los demás sistemas anteriormente citados.

1.1.2 Envío de trabajos

Indiferentemente del sistema de encolamiento usado, las tareas (jobs) enviadas a ejecutar en un clúster tienen un ciclo de vida común el cual permite mantener un orden en la gestión y procesamiento de estas

tareas (6). A continuación se enumeran en orden los pasos básicos por los que transita la tarea enviada al clúster:

1. Envío de la tarea.
2. El *frontend*[4] almacena la tarea en su base de datos y notifica al planificador.
3. El planificador asigna la tarea a una instancia de una cola.
4. El frontend envía la tarea al host correspondiente o según las especificaciones del usuario puede ser enviado a más de un nodo.
5. Los nodos encargados de la ejecución almacenan la tarea en su base de datos.
6. Se ejecuta la tarea en cada nodo correspondiente y se espera por su terminación.
7. Una vez terminada la ejecución se informa al frontend y se elimina de la base de datos del nodo.

1.1.3 Sistemas por lotes (7)

Se le llama sistema por lotes al trabajo con un conjunto de computadoras y otros recursos (redes, sistemas de almacenamiento, servidores de licencias, etc.) que operan bajo la idea de que la unión de las partes es más que el todo por separado. Algunos sistemas por lotes están compuestos por pocos equipos que ejecutan tareas en un solo procesador y siendo gestionados por los propios usuarios. Otros sistemas tienen miles de máquinas que ejecutan tareas de diferentes usuarios de forma simultánea, mientras que le dan seguimiento, al mismo tiempo, al acceso a los equipos y a los sistemas de almacenamiento.

Los sistemas por lotes se componen de cuatro elementos diferentes: Frontend, Nodos Interactivos, Nodos de cómputo y Recursos.

Frontend

Un sistema por lotes tiene un nodo principal donde se ejecuta el servidor que mantendrá el control del resto de los nodos. Dependiendo de las necesidades de los sistemas, un nodo maestro se puede dedicar a esta tarea solamente, o también puede cumplir con las funciones de los otros componentes.

Nodos Interactivos

Los nodos interactivos proporcionan un punto de entrada al sistema para los usuarios gestionar sus tareas. Para estos nodos, los usuarios pueden enviar y dar seguimiento a sus tareas.

[4] Nodo máster encargado de la gestión de los demás nodos del clúster.

Nodos de cómputo

Los nodos de cómputo son los caballos de batalla del sistema. Su función es ejecutar los trabajos enviados. En cada nodo de cómputo, corre un cliente de ejecución para iniciar, eliminar y administrar las tareas. Estos nodos se comunican con el frontend para informar de las soluciones y acciones realizadas sobre las tareas. Dependiendo de las necesidades de los sistemas, un nodo de cómputo puede hacer función de frontend (nodo maestro).

Recursos

Algunos sistemas están organizados con el propósito expreso de la gestión de una colección de recursos. Los recursos pueden incluir redes de alta velocidad, sistemas de almacenamiento, etc. La disponibilidad de estos recursos es limitada y debe ser gestionada de forma inteligente para promover la equidad y la mejor utilización.

1.2 Tecnologías Web

El éxito de las aplicaciones web se basa en dos puntos fundamentales: el protocolo HTTP y el lenguaje HTML. Uno permite una implementación simple y sencilla de un sistema de comunicaciones, mientras el otro proporciona un mecanismo de composición de páginas enlazadas de forma fácil y altamente eficiente (8). Ambos son los componentes fundamentales del modelo cliente-servidor

1.2.1 Apache Tomcat

Es un servidor web, de código abierto que une la tecnología Java Servlet y Java Server Pages. Es desarrollado en un entorno abierto y participativo y publicado bajo la licencia Apache 2.0. Está destinado a ser una colaboración de los mejores desarrolladores de su clase de todo el mundo. En él se pueden montar aplicaciones de misión crítica a través de la Web (9). Es un servidor HTTP y un contenedor de servlets (clases de Java), cargándolos y ejecutándolos de forma dinámica. Permite montar servicios web mediante Axis (10) (11).

1.2.2 Portlet

Es un componente Web basado en la tecnología Java que procesa los pedidos del usuario y genera un contenido dinámico final (12), el contenido generado por un Portlet es llamado fragmento y su ciclo de vida

es manejado por el contenedor de Portlets. Puede declarar los eventos que quiere emitir y los que desea recibir. El contenedor de Portlet actuará como intermediario y distribuirá los eventos en consecuencia.

1.2.3 Liferay Portal

Es un portal de gestión de portlets, de código abierto e implementado en Java (13). Se puede desplegar en servidores como Apache Tomcat, es multiplataforma lo que le permite ejecutarse en cualquier sistema operativo. Liferay es un proyecto de código abierto que usa licencia LGPL. Provee la personalización del entorno de usuario mediante plantillas y temas que pueden ser instalados a partir de un archivo con extensión .WAR.

1.2.4 Portal de Servicios Bioinformáticos de la UCI

Es un portal desarrollado sobre el contenedor de Porlets, Liferay Portal (14), que provee un ambiente amigable para la definición y la ejecución de análisis en la esfera de la bioinformática. Su objetivo es proporcionarle a los especialistas bioinformáticos el acceso a recursos computacionales, los cuales muchas veces carecen en sus centros de trabajo o de estudio (15).

1.3 Lenguajes de programación

Los lenguajes de programación son herramientas que facilitan la tarea de crear programas y software ya que disponen de sintaxis adecuadas que permiten ser leídas y escritas por las personas (16) y a su vez le permiten la interacción con el hardware y el software presente en la máquina (17).

1.3.1 Java

Es un lenguaje orientado a objetos desarrollado bajo una licencia libre. Fue desarrollado por la Sun Microsystems con la idea de utilizarse para el desarrollo de aplicaciones web, aunque luego se percataron de la amplia gama de posibilidades que tiene sobre las aplicaciones de escritorio. Los programas desarrollados con Java son independientes de la arquitectura de la computadora donde se ejecutan, así como que también son independientes del sistema operativo que los ejecuta ya que el código Java no se ejecuta sobre el sistema operativo sino sobre una máquina virtual (JDK). Permite aplicaciones bajo el esquema de Cliente-Servidor (16).

1.4 Entornos de Desarrollo Integrado

Los Entornos de Desarrollo Integrado (IDE por sus siglas en inglés) son un conjunto de programas que se ejecutan a partir de una única interfaz de usuario. En el caso de los IDE's para programadores incluyen a menudo un editor de texto, un compilador y un depurador, los cuales se activan y funcionan a partir de un menú común (18).

1.4.1 Eclipse

Desarrollado por IBM y entregado a la Fundación Eclipse sin fines lucrativos para ser utilizado como plataforma de código abierto y disponible para todos los sistemas operativos. Además de proporcionar un entorno de desarrollo amigable, automatiza muchas funciones a la hora del completado del código que de otra manera el desarrollador tendría que poner a mano. Cuenta con el apoyo de muchos observadores que dicen que es la herramienta clave para el desarrollo de la industria del software. Es fácil de usar y mucho más fácil de integrar sus distintos componentes y herramientas (19).

Como es una plataforma IDE, existen plugins para la mayoría de los lenguajes conocidos como por ejemplo: Java, C/C++, PHP, Cobol, etc. (20).

1.5 Marcos de trabajo

En el desarrollo de software, un framework (marco de trabajo), es una estructura conceptual y tecnológica de soporte definido, normalmente con módulos de software concretos, con base a la cual otro proyecto de software puede ser más fácilmente organizado y desarrollado. (21)

1.5.1 Spring

Spring Framework proporciona una amplia configuración de la programación y el modelo de las modernas aplicaciones empresariales, basadas en Java. Un elemento clave de Spring es el apoyo de infraestructura a nivel de aplicación. Se centra en la "fontanería" de las aplicaciones de forma que los equipos pueden centrarse en la lógica del negocio a nivel de aplicación, sin ataduras innecesarias a los entornos de despliegue específicos (22).

1.5.2 Axis2

Es un middleware, que permite la comunicación con otras aplicaciones, que permite montar servicios sobre la Web de Apache. Su arquitectura se basa en desarrollar sobre un núcleo simple y extensible que proporciona las abstracciones básicas para el resto del sistema (23).

1.6 Metodología de desarrollo de software

Una metodología de desarrollo de software se refiere a un framework que es usado para estructurar, planear y controlar el proceso de desarrollo en sistemas de información. Los framework para metodología de desarrollo de software consisten en una filosofía de desarrollo de programas de computación con el enfoque del proceso de desarrollo de software y en herramientas, modelos y métodos para asistir al proceso de desarrollo de software. (24)

1.6.1 OpenUp

Su objetivo es asegurar la producción de software de calidad dentro de los plazos propuestos. Dirigido por casos de usos, centrado en la arquitectura, iterativo e incremental. Desarrollado y mantenido por la comunidad libre bajo licencia libre (25). Es un proceso mínimo, está centrado en la arquitectura y solamente es incluido el contenido fundamental para reducir al mínimo los riesgos. Es completo, puede ser manifestado como todo el proceso para construir un sistema. Es extensible, se puede agregar o se puede adaptar según lo vayan requiriendo los sistemas, en otras palabras, es ágil. Proporciona una comprensión detallada del proyecto, beneficiando tanto a clientes como a desarrolladores sobre los productos a entregar. Es la metodología más utilizada por desarrolladores de alto nivel en casi todo el mundo por sus cualidades administrativas (26).

1.7 Lenguaje de Modelado

1.7.1 UML

Lenguaje de Modelado Unificado (UML por sus siglas en Inglés) es la sucesión de una serie de métodos de análisis y diseño orientadas a objetos que aparecen a fines de los 80's y principios de los 90's. Una de las metas principales de UML es avanzar en el estado de la integración institucional proporcionando herramientas de interoperabilidad para el modelado visual de objetos (27). Se utiliza para definir un

sistema, para detallar los artefactos en el sistema, para documentar y construir. Es el lenguaje en el que está descrito el modelo.

1.8 Herramientas CASE de modelado

Ingeniería de Software Asistida por Computadoras (CASE por sus siglas en Inglés) es la aplicación de métodos y técnicas (28) enfocadas a ayudar a los analistas de software desde el inicio del ciclo de vida de un software hasta que termina con la última iteración del mismo proporcionándoles diagramas de casos de usos y de interfaces y autogenerando código de programación a partir de estos diagramas (29).

1.8.1 Visual Paradigm para UML

Es una herramienta UML profesional, libre, que soporta el ciclo de vida completo del desarrollo del software: análisis y diseño orientado a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación que sirven de puente entre los arquitectos, analistas y diseñadores del sistema haciendo el trabajo más fácil y dinámico. También automatiza tareas tediosas que pueden distraer a los diseñadores. La navegación es intuitiva entre el código y el modelo visual. Es un poderoso generador de informes. Mantiene la demanda en tiempo real. Contiene un ambiente modelador visual superior y un sofisticado diseño de diagramas (30).

1.9 Conclusiones.

En el presente capítulo se explicaron los principales fundamentos teóricos de la gestión de Clústeres Beowulf lo cual forma parte de la información necesaria para dar solución a la problemática planteada. Se seleccionó para la solución el uso de un nuevo estándar, Portlets debido a las ventajas que brinda. Para desarrollar dicho estándar, después de analizar las diferentes técnicas y herramientas, se seleccionó como contenedor de Portlets el Liferay Portal, la metodología de desarrollo OpenUp, el lenguaje de modelado visual UML, la herramienta Case para el modelado Visual Paradigm, el lenguaje de programación Java y finalmente como IDE de desarrollo el Eclipse.

CAPÍTULO 2. CARACTERÍSTICAS DEL SISTEMA.

En el presente capítulo se ofrece una breve descripción de la solución propuesta en vista de solucionar el problema; así como el modelo de dominio y los requisitos funcionales y no funcionales. Se realiza una representación de los diagramas de casos de uso, su descripción y los actores que intervienen en ellos.

2.1 Descripción del Sistema

El portlet para la gestión del clúster posibilitará al investigador gestionar las tareas y usuarios del clúster Beowulf del centro DATEC. Se implementará una interfaz de acceso al clúster para brindar servicios posteriores hasta llegar a realizar la gestión total del clúster mediante interfaces visuales.

2.2 Modelo de dominio

Un Modelo de Dominio (MD) se aplica cuando no es posible realizar el modelo del negocio. Este modelo debe ser muy delgado, capturando las entidades principales de negocio y las relaciones entre ellos. El mismo representa un aparte visual de las clases y sus interacciones con el entorno del proyecto (31). El diagrama (ver Figura 1) representa los objetos relacionados con los principales conceptos que se emplearán en este trabajo utilizando notación UML de modelado de datos.

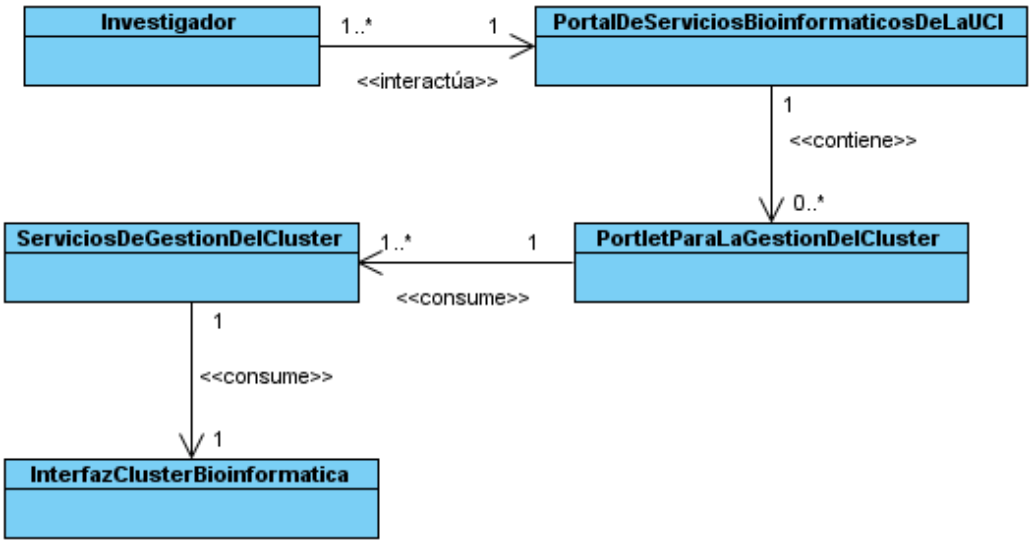


Figura 1. Diagrama de Modelo de Dominio.

2.2.1 Definición de clases del modelo de dominio

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

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

PortletGestiónMonitorizaciónDelClúster: Portlet que le permite al especialista la gestión de las tareas y recursos del clúster.

ServicioMonitoreoGestiónDelClúster: Servicio que brinda la posibilidad de acceder al clúster para gestionar sus tareas y usuarios.

InterfazClústerBionformática: Interfaz a línea de comando mediante la cual se accede directamente al clúster.

2.3 Requerimientos del sistema

Los requerimientos del software son las capacidades o condiciones que el sistema debe cumplir o alcanzar. A continuación se muestran los requisitos, separados en funcionales y en no funcionales.

2.3.1 Requisitos Funcionales

Los requerimientos funcionales surgen de la entrevista con el cliente, el cual plantea de forma clara las necesidades que considera debe satisfacer el sistema. A continuación se enumeran.

RF1 Autenticar Usuario. El usuario debe ser capaz de autenticarse en el sistema, así podrá tener acceso a las funcionalidades del mismo.

RF2 Añadir nueva tarea al clúster: Este software debe ser capaz de permitir añadir una nueva tarea al clúster.

RF3 Eliminar una tarea asignada al clúster: Este software debe permitir eliminar una tarea asignada al clúster.

RF4 Listar tareas asignadas al clúster: Este software debe permitir listar las tareas asignadas al clúster.

RF5 Verificar estado de las tareas: Este software debe permitir chequear el estado de cada tarea introducida al clúster.

RF6 Descargar las soluciones de las tareas: Este software debe permitir descargar los resultados de las tareas finalizadas en el clúster.

RF7 Adicionar nuevos usuarios al Clúster. Este software debe permitir a los usuarios con permisos administrativos añadir nuevos usuarios al clúster.

RF8 Eliminar usuarios del Clúster. Este software debe permitir a los usuarios con permisos administrativos eliminar usuarios del clúster.

RF9 Editar usuarios del Clúster. Este software debe permitir a los usuarios con permisos administrativos editar los usuarios del clúster.

RF10 Listar los usuarios del clúster. Este software debe permitir a los usuarios con permisos administrativos listar los usuarios existentes en el clúster.

2.3.2 Requisitos no funcionales.

Los requerimientos no funcionales son propiedades o cualidades que el sistema debe tener. Son importantes para que los usuarios valoren las características no funcionales del sistema. A continuación se muestran.

- **Seguridad**

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

- **Rendimiento**

  El sistema está concebido para un ambiente cliente-servidor por lo que la rapidez del servicio dependerá del volumen de la información que necesite procesar el clúster para cada petición.

- **Usabilidad**

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

- **Apariencia o interfaz externa**

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

- **Portabilidad**

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

- **Soporte**

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

- **Software**

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

- Navegador Web estándar con capacidad de interpretación de Java Script, CSS compatible con la W3Cejemplo Netscape, optimizado para Mozilla 1.7 o superior o FireFox 0.9.3 o superior.

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

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

- **Hardware**

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

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

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

- Los servidores deberán contar con un microprocesador Intel Pentium IV o superior, a 3.0 GHz o superior, con un mínimo de 1GB de memoria RAM, un disco duro con capacidad disponible

de almacenamiento de 40GB y una tarjeta de red para poder brindar el servicio utilizando la red.

- **Legales**

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

- **Restricciones de diseño e implementación**

El análisis y el diseño de la aplicación serán basados en la Metodología OpenUp con el uso del lenguaje de modelado UML. Como herramienta CASE será usada Visual Paradigm para el modelado y como lenguaje de programación Java (JSP).

2.4 Actores del sistema

Los actores del sistema son personajes o entidades que guardan relación con los casos de uso y que le demanda una funcionalidad.

Tabla 1. Actores del sistema. En la tabla se especifican los actores que interactúan con el sistema.

Actor	Descripción
Investigador	Representa al usuario que va a hacer uso del sistema, con posibilidades de realizar interacción con todas sus funcionalidades.

2.4.1 Diagrama de casos de uso del Sistema

Los diagramas de casos de uso se utilizan para la ilustración de los requisitos del sistema al mostrar cómo reacciona a eventos que se producen en su contorno o en él mismo. Muestra la relación que existe entre los actores y los casos de uso en un sistema. Especifican la comunicación y el comportamiento de un sistema (Figura 2).

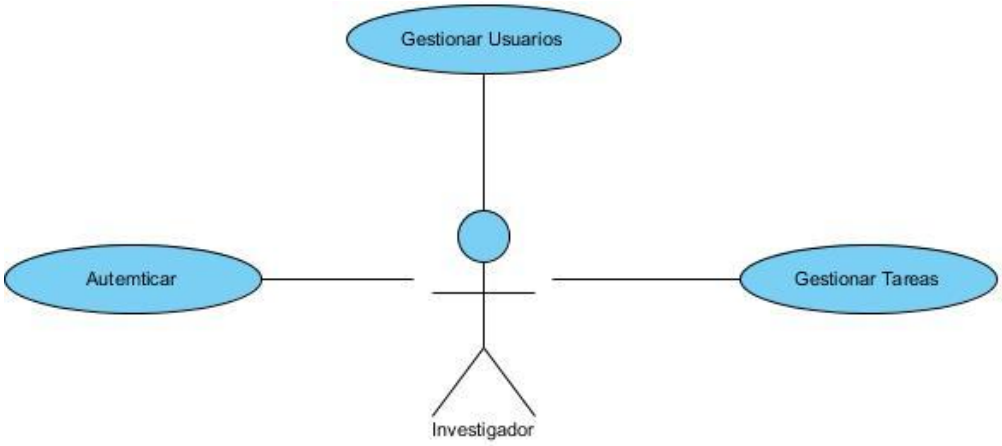


Figura 2. Diagrama de casos de uso del Sistema.

2.5 Descripción de los casos de uso del Sistema

2.5.1 Caso de uso Gestionar Tareas

Tabla 2. Descripción del CU "Gestionar Tareas". En la tabla se especifican las descripciones de todas las secciones del CU Gestionar Tareas con sus flujos alternos.

Caso de Uso	Gestionar Tareas.
Actor	Investigador.
Propósito	Crear y modificar tareas en el clúster.
Resumen	El caso de uso se inicia cuando el investigador va a realizar alguna de las siguientes operaciones relacionadas con la gestión de tareas: Adicionar nueva tarea. Listar tareas del investigador existentes en el clúster. Verificar el estado de una tarea. Descargar el resultado de una tarea. Eliminar una tarea. Por último, se finaliza el caso de uso cuando el sistema actualiza sus datos.

Referencia	RF3, RF4, RF5, RF6, RF7.
Precondición	Debe estar autenticado.
Prioridad	Crítico.

Flujo Normal de Eventos	
Acción del Autor	**Respuesta del Sistema**
1. El caso de uso se inicia cuando el investigador selecciona una de las siguientes opciones: Adicionar nueva tarea. Listar tareas del investigador existentes en el clúster. Verificar el estado de una tarea. Descargar el resultado de una tarea. Eliminar una tarea.	2. El sistema en dependencia de la opción seleccionada realizará lo siguiente: Si seleccionó Adicionar nueva tarea, ir a sección “Adicionar nueva tarea”. Si seleccionó Listar tareas del investigador existentes en el clúster ir a sección “Listar tareas del investigador existentes en el clúster”. Si seleccionó Verificar el estado de una tarea ir a sección “Verificar el estado de una tarea”. Si seleccionó Descargar el resultado de una tarea ir a sección “Descargar el resultado de una tarea”. Si seleccionó Eliminar una tarea ir a sección “Eliminar tarea”.

Sección 1 “Adicionar nueva tarea”	
Acción del Actor	**Respuesta del Sistema**
1.1 El investigador selecciona la opción de insertar nueva tarea, para ello en la interfaz principal presiona el botón “Adicionar nueva Tarea”.	1.2 El sistema muestra al investigador un formulario donde este podrá insertar los parámetros necesarios para adicionar una nueva tarea.
1.3 El investigador inserta los datos necesarios y presiona el botón “Enviar tarea”.	1.4 El sistema comprueba que los datos insertados sean correctos y añade la nueva tarea a la lista de tareas existentes.
Flujos Alternos de la sección 1 “Adicionar nueva tarea” del paso 1.4	

Acción del Actor	Respuesta del Sistema
	1.4 Si los valores insertados por el investigador no son correctos el sistema no adiciona la tarea y muestra un mensaje advirtiendo al investigador de lo ocurrido. Finaliza el caso de uso.
Sección 2 "Listar tareas del investigador en el clúster"	
Acción del Actor	**Respuesta del Sistema**
2.1 El investigador introduce sus credenciales en el sistema.	2.2 El sistema muestra en su interfaz principal la lista de tareas correspondientes al investigador. Fin del caso de uso.
Sección 3 "Descargar el resultado de una tarea"	
Acción del Actor	**Respuesta del Sistema**
3.1 El investigador selecciona el botón correspondiente a descargar resultados de la tarea que desee.	3.2 El sistema muestra al investigador una interfaz donde este puede seleccionar descargar resultados.
3.3 El investigador presiona el botón "descargar" y selecciona el destino donde se guardará el mismo.	3.4 El sistema guarda el archivo de los resultados en donde el investigador especificó. Fin del caso de uso.
Flujos Alternos de la sección 3 "Descargar el resultado de una tarea" del paso 3.3	
Acción del Actor	**Respuesta del Sistema**
3.3 El investigador selecciona no guardar los resultados en el paso 3.3	3.4 El sistema vuelve a mostrar la lista de tareas. Finaliza el caso de uso.
Sección 4 "Eliminar tarea"	
Acción del Actor	**Respuesta del Sistema**
4.1 El investigador selecciona la opción eliminar tarea.	4.2 El sistema muestra un mensaje de confirmación para que el investigador

	especifique que quiere realizar esa acción.
4.3 El investigador selecciona la opción de eliminar.	4.4 El sistema elimina la tarea correspondiente y muestra nuevamente la lista de tareas actualizada. Finaliza el caso de uso.
Flujos Alternos de la sección 3 "Eliminar tarea" del paso 4.3	
Acción del Actor	**Respuesta del Sistema**
4.3 el investigador selecciona "cancelar" eliminar tarea.	4.4 El sistema no elimina la tarea y muestra la lista de tareas correspondientes al investigador. Fin del caso de uso.
Sección 5 "Verificar estado de una tarea"	
Acción del Actor	**Respuesta del Sistema**
5.1 El investigador introduce sus credenciales en el sistema.	5.2 El sistema muestra en la interfaz principal una lista de tareas en las cuales se puede visualizar si dicha tarea está terminada o no.
Poscondiciones	El investigador debe encontrarse en la interfaz principal del sistema.

2.6 Conclusiones.

En este capítulo se expusieron los requisitos funcionales con los que debe contar la aplicación para que cuente con la calidad requerida, sea confiable, usable, con la ayuda de los requisitos no funcionales, así como los casos de uso del sistema y el actor. Se definieron 10 requisitos funcionales agrupados en 3 casos de uso. Los casos de uso serán de utilidad para cumplir con exactitud la siguiente fase del desarrollo del software, el proceso de diseño e implementación.

CAPÍTULO 3. DISEÑO DE LA APLICACIÓN.

Luego de haber comprendido los conceptos y los requisitos del sistema en el capítulo anterior, se procede a diseñar la solución propuesta. Este capítulo brinda un acercamiento a la implementación del sistema, para ello, se construyen los diagramas de interacciones del mismo, así como el modelo de clases del diseño, los cuales proporcionan la base para proceder a la etapa de construcción.

3.1 Arquitectura de software

En el desarrollo cualquier software, tanto la calidad de diseño como la asignación de responsabilidades pueden mostrar grandes variaciones. La mala toma de decisiones en este sentido puede dar como resultado aplicaciones con un funcionamiento deficiente y de una trabajosa manipulación. Una vía inteligente para disminuir estos errores de fundamenta en los principios de una buena arquitectura de software.

La arquitectura de software define la estructura de un sistema. Esta estructura se forma uniendo componentes, módulos o piezas de código que surgen de la abstracción, cumpliendo funciones específicas e interactuando entre sí con un comportamiento definido. Los componentes se organizan de acuerdo a ciertos criterios, que representan decisiones de diseño. En los últimos años la arquitectura de software se ha debatido en dos corrientes fundamentales: los estilos arquitectónicos y los patrones arquitectónicos.

3.1.1 Estilos y patrones arquitectónicos

Un estilo arquitectónico es como una familia de sistemas de software en términos de un patrón de organización estructural, que define un vocabulario de componentes y tipos de conectores y un conjunto de restricciones de cómo pueden ser combinadas (32). Para muchos estilos puede existir uno o más modelos que especifiquen cómo determinar las propiedades generales del sistema.

Existen disímiles categorías para los estilos arquitectónicos, que incluyen varios estilos, como por ejemplo: Llamada y retorno, que incluye la arquitectura orientada a objetos, arquitecturas basadas en componentes y arquitectura en capas y la categoría Peer to peer que contiene estilos de arquitectura basada en eventos, arquitecturas orientada a servicios y arquitectura basada en recursos. Existen otros estilos como Tuberías y filtros, Código móvil y Arquitectura centrada en datos.

Los términos estilo y patrón arquitectónicos son a menudo confundidos, ya que están muy ligado. Un patrón arquitectónico se define como un conjunto de restricciones en una arquitectura que describe sus elementos y los tipos de relaciones entre ellos (33). Aunque no se precise de una definición estándar para estos términos, sean estilos o patrones arquitectónicos, sus aplicaciones en el desarrollo de software constituyen un elemento elemental.

3.1.1 Patrones de diseño

Al momento de elaborar los diagramas de interacción es importante asignar correctamente las responsabilidades. Para ello, podemos auxiliarnos de los patrones de diseño, entre los que se pueden mencionar los GRASP, más conocidos como patrones de asignación de responsabilidades, los patrones GoF y el Modelo Vista Controlador (MVC).

Los patrones GRASP describen los principios de la asignación de responsabilidades a objetos, expresados en forma de patrones (34). A continuación se enumeran algunos de los más usados.

1. **Experto**

   Asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad.

2. **Creador**

   Explica que clase es la encargada de crear objetos, en determinados escenarios de ejecución.

3. **Bajo acoplamiento**

   Asignar una responsabilidad para mantener bajo acoplamiento. Explica cómo dar soporte a una dependencia escasa y a un aumento de la reutilización.

4. **Alta Cohesión**

   Asignar una responsabilidad de modo que la cohesión siga siendo alta. Explica cómo mantener la complejidad dentro de límites manejables.

5. **Controlador**

   Asignar la responsabilidad del manejo de los eventos de un sistema a una clase que represente un sistema global.

Los patrones GoF complementan a los patrones GRASP y en ocasiones se puede encontrar una contraposición entre este tipo de patrones, e incluso, podría inferirse que algunos patrones GoF son

variantes de los patrones GRASP, es por ello que la decisión de utilizar uno u otro debe tomarse con precaución y aplicarse sólo en el ámbito necesario. Entre los patrones GoF más utilizados se pueden citar el de Agente y Agente Remoto, Fachada y Agente Dispositivo y el Comando.

El patrón Agente explica el comportamiento cuando no se desea o no es posible acceder directamente a un componente, para ello este patrón sugiere definir una clase sustituta de software que represente al componente y asignarle la responsabilidad de comunicarse con el componente real. Por su parte el patrón Agente Remoto es un caso especial del patrón Agente (34). Este define como el sistema debe comunicarse con un componente situado en otro espacio o contexto a través de una clase de software local que represente al componente externo y asignarle la responsabilidad de contactar al componente real.

El patrón Fachada es generalmente utilizado para resolver problemas de integración, y se aplica cuando se requiere una interfaz común de comunicación con un conjunto de interfaces o funciones de otro subsistema, en este sentido Fachada define una sola clase que unifique las interfaces y le asigna la responsabilidad de colaborar con el subsistema. El patrón Agente Dispositivo es una especificidad de Fachada cuando el subsistema que se quiere integrar es un dispositivo externo.

Uno de los patrones de diseño generalizado en las aplicaciones actuales es el MVC, aunque en muchos textos este patrón se define como un patrón arquitectónico. El MVC separa el sistema en tres capas fundamentales, una capa de modelo, que representa los datos relacionados con la aplicación, una capa de vista o presentación, encargada de representar los datos del modelo al usuario, mediante la cual se permitirá manipular la información y una capa controladora que procesa las peticiones que vienen desde la capa de presentación y donde está encapsulada la lógica del negocio (Figura 3).

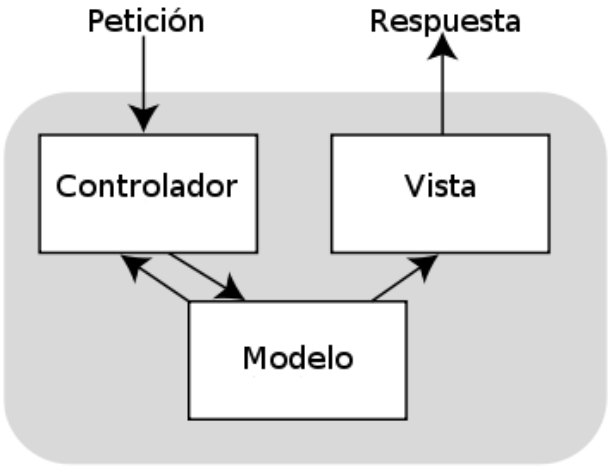


Figura 3. Diagrama del patrón Modelo Vista Controlador (MVC).

3.2 Diagramas de interacción

Para obtener un buen resultado en la resolución de los requisitos funcionales identificados es importante la elaboración de diagramas de interacción, que muestran gráficamente cómo los objetos se comunican entre sí con el fin de cumplir con las funcionalidades.

El lenguaje de modelado UML define dos tipos de diagramas de interacción, ambos sirven para expresar interacciones semejantes o idénticas de mensaje, ellos son: los diagramas de secuencia y los diagramas de colaboración. Los diagramas de secuencia describen las interacciones en una especie de formato de cerca o muro, y se representan en el tiempo mediante líneas temporales en sentido vertical, mientras que los de colaboración describen las interacciones entre los objetos en un formato de grafo o red.

Los diagramas de interacción constituyen uno de los artefactos más importantes que se generan en el análisis y en el diseño, orientados a objetos (34). Es a partir de este planteamiento que se definen en las siguientes secciones este tipo de diagramas.

3.2.1 Diagramas de secuencia y colaboración

Los diagramas de secuencia muestran gráficamente los eventos que fluyen desde los actores hacia el sistema. Es conveniente crear un diagrama de secuencia por cada operación del sistema, y enfatizar la secuencia de los eventos en el flujo normal de eventos de los CU. A continuación se tratará el diagrama de secuencia del CU: “Gestionar taras del clúster”.

El diagrama de secuencia de la Figura 4 muestra la comunicación de los componentes del software. Se define una Clase Controladora (CC) llamada ControladorFrontal.xml encargada de gestionar todas las peticiones del usuario y de determinar el controlador que le dará respuesta a dicha petición, en este caso controllerPrincipal.java. De esta manera se aplican patrones como el Controlador de GRASP y el Comando de los GoF, también están los patrones Bajo acoplamiento y Alta cohesión.

Por otro lado se identifican una serie de métodos o funciones en las clases controladoras, las cuales deben ser capaz, entre otras cosas, de mostrar formularios y validar los datos que provienen de los mismos.

En la Figura 5 se muestra el diagrama de colaboración del CU “Gestionar tareas del clúster” para la sección “Añadir tarea”.

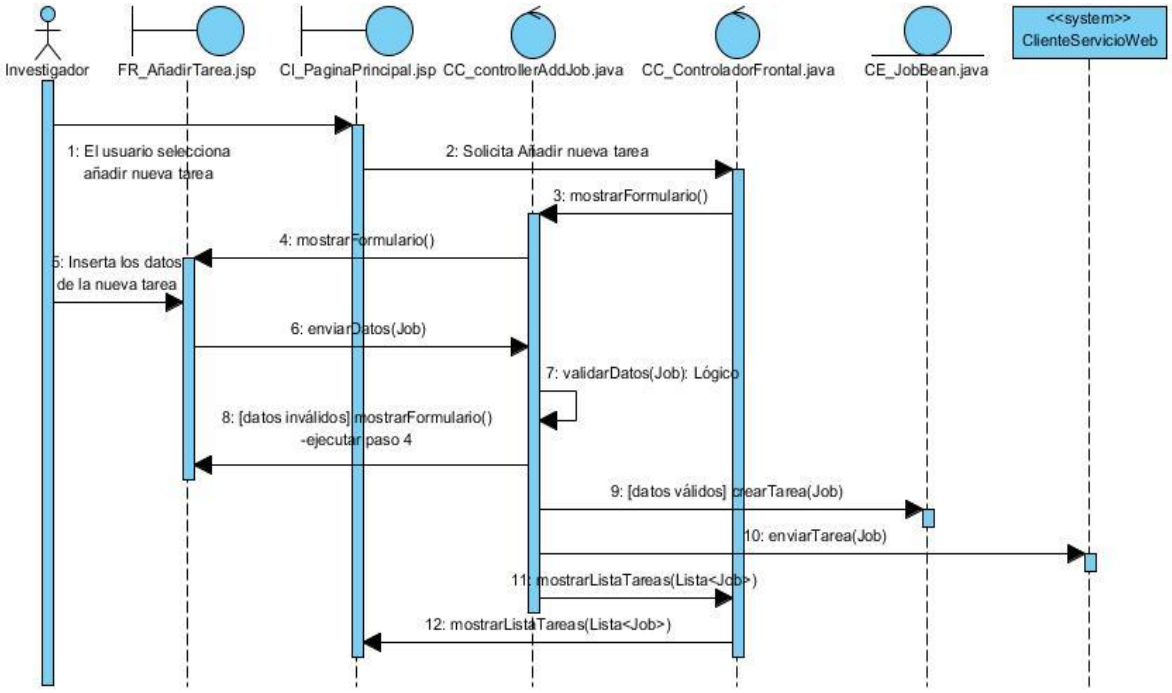


Figura 4. Diagrama de secuencia del Caso de uso Añadir Tarea.

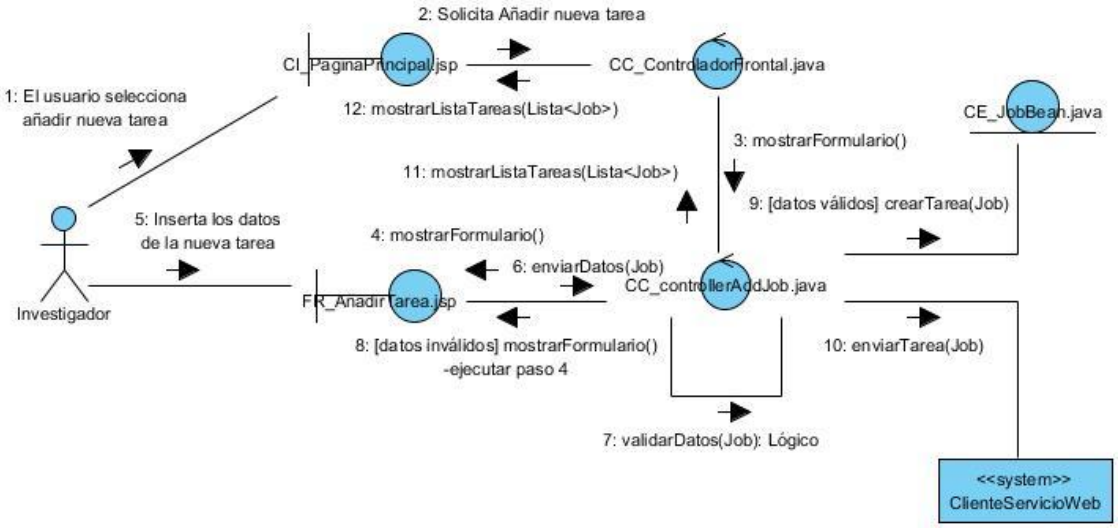


Figura 5. Diagrama de colaboración del Caso de uso Añadir Tarea.

3.3 Modelo del diseño

Ya elaborados los diagramas de interacción, se pueden identificar las clases del software que nos llevarán a la solución del sistema y complementar estas con el diseño. Un diagrama de clases del diseño, contiene las definiciones de las entidades del software.

En el diagrama de clases del diseño para el CU “Añadir Tarea” (Figura 6), se identifican tres paquetes funcionales: Presentación, Lógica de aplicación, los que corresponden a las capas del patrón de diseño MVC, y por último el Cliente de Servicios Web. Además se puede apreciar la distribución de los objetos dentro de estos paquetes, separando el sistema en varias capas, aplicando el patrón arquitectura en capas.

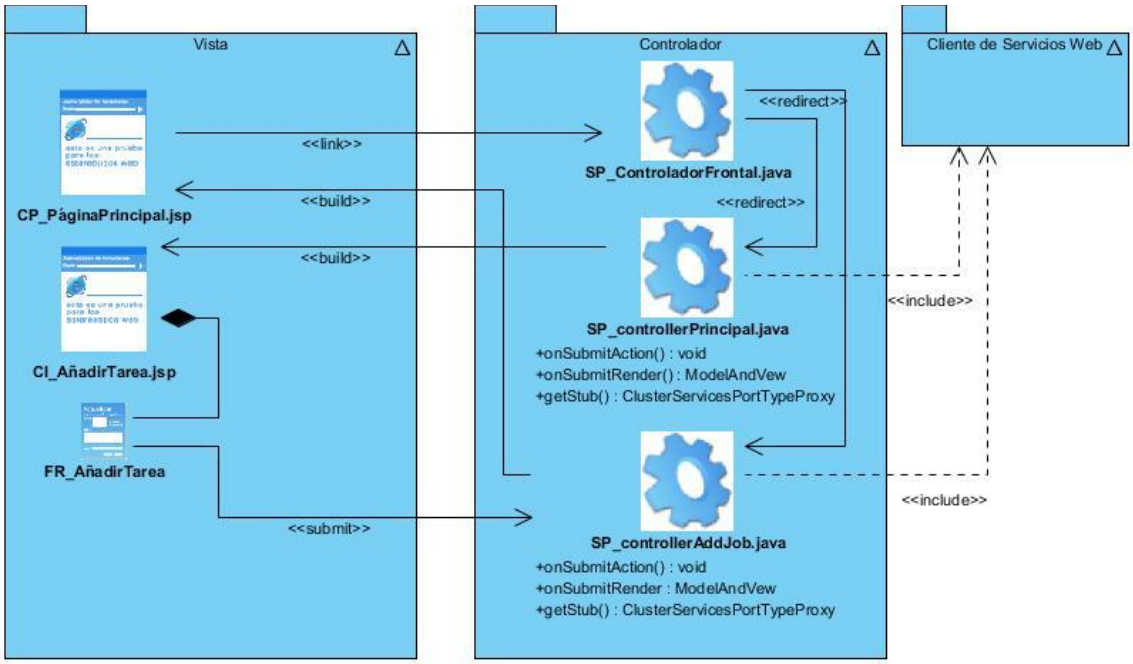


Figura 6. Diagrama de Clases del diseño.

El diagrama de clase del diseño correspondiente al paquete Cliente de Servicios Web (Figura 7) muestra los componentes encargados de realizar las llamadas a las funciones remotas del sistema.

En este paquete la interfaz ClusterServicesPortType.java declara las funciones que invocan los servicios mientras que la clase ClusterServicesPortTypeProxy.java define las mismas. El subsistema Servicios Web Clúster representa los servicios albergados en el servidor de servicios web. En este diagrama se manifiesta el patrón Agente Remoto de la familia GoF a través de la clase ClusterServicesPortTypeProxy.java.

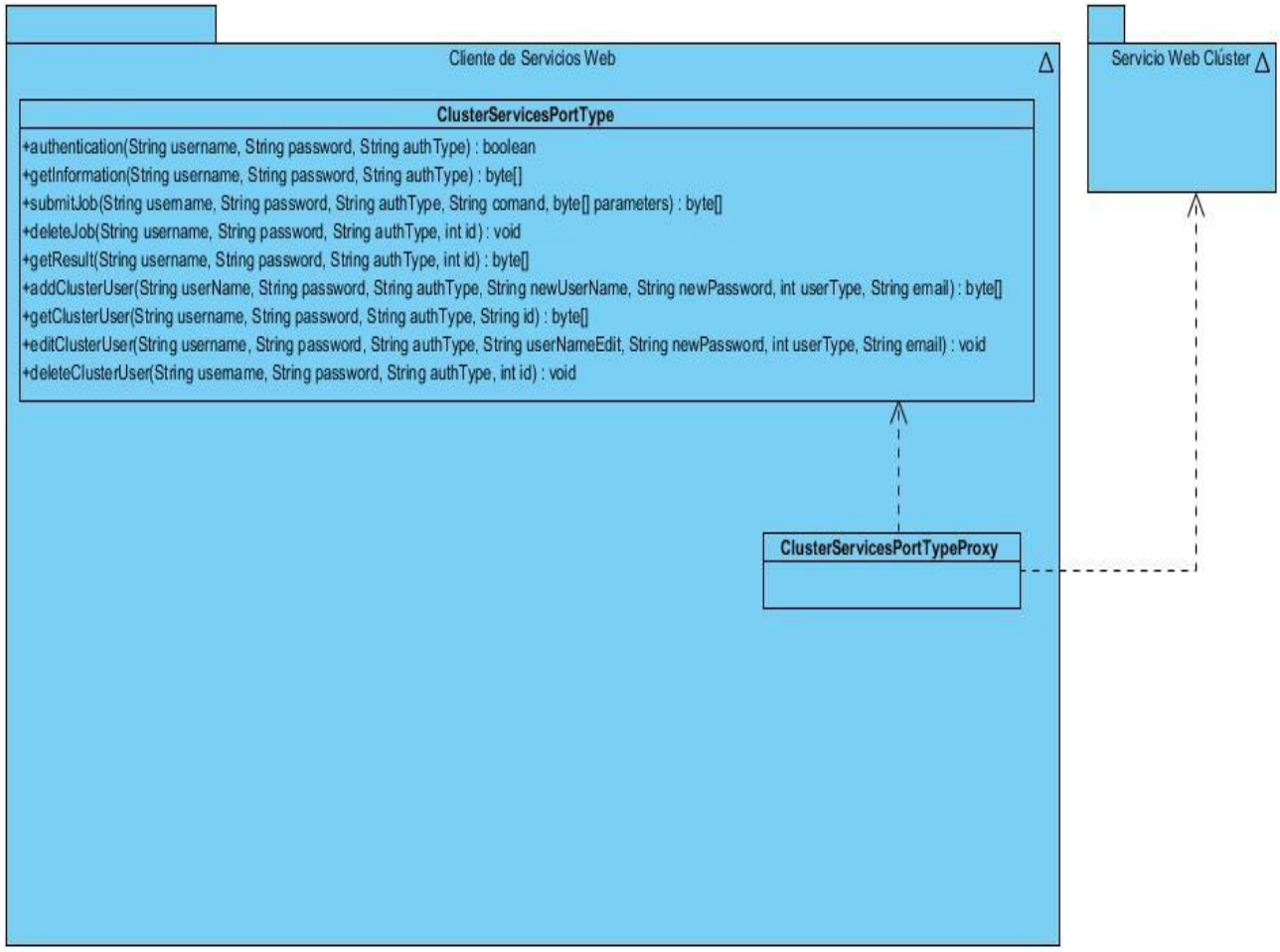


Figura 7. Diagrama de clases del diseño del paquete Cliente de Servicios Web.

3.4 Conclusiones

En el presente capítulo se definieron los diagramas de interacción del sistema para el CU de mayor impacto en la arquitectura: “Gestionar Tareas”, específicamente para la sección “Añadir tarea”. Para la elaboración de los mismos se analizaron diferentes patrones arquitectónicos y patrones de diseño que posibilitaron definir una mejor arquitectura, entre estos se definieron la arquitectura cliente-servidor y orientada a servicios, así como el MVC, Alta cohesión y Bajo acoplamiento.

CAPÍTULO 4. IMPLEMENTACIÓN Y PRUEBA.

Una vez concluido el modelo del diseño, procedemos a la construcción del sistema y ya construido procedemos a la realización de pruebas para verificar el correcto funcionamiento de los requisitos funcionales de nuestra aplicación. En el presente capítulo se construye el modelo de implementación correspondiente; los diagramas de componentes de los requisitos más importantes, de la misma forma se determinan los tipos de pruebas a realizar y los casos de pruebas que serán aplicados al sistema.

4.1 Modelo de Implementación

El modelo de despliegue se crea durante las últimas actividades del flujo de trabajo de diseño y brinda la descripción de la distribución física del sistema. Se utiliza como entrada fundamental en las actividades de diseño e implementación. Por otra parte el diagrama de componentes estructura el modelo de implementación y muestra las relaciones entre los elementos de implementación. Estos diagramas conforman el modelo de implementación, que describe como se implementan los elementos del diseño en términos de componentes tales como ficheros de código fuente, ejecutables, librerías y documentos.

4.1.1 Diagrama de componentes

El diagrama de componentes se representa como un conjunto de componentes de software unido por medio de las relaciones de dependencia. Entre los componentes se encuentran las clases, interfaces, librerías y componentes ejecutables. Los diagramas de componentes se utilizan para modelar código fuente, versiones ejecutables, bases de datos físicas entre otros. El diagrama de componentes muestra un conjunto de ficheros relacionados entre sí para lograr completar una funcionalidad del sistema.

En la Figura 8 se muestra el diagrama de componentes del CU: “Gestionar Tareas” para la sección *“Añadir Tarea*”.

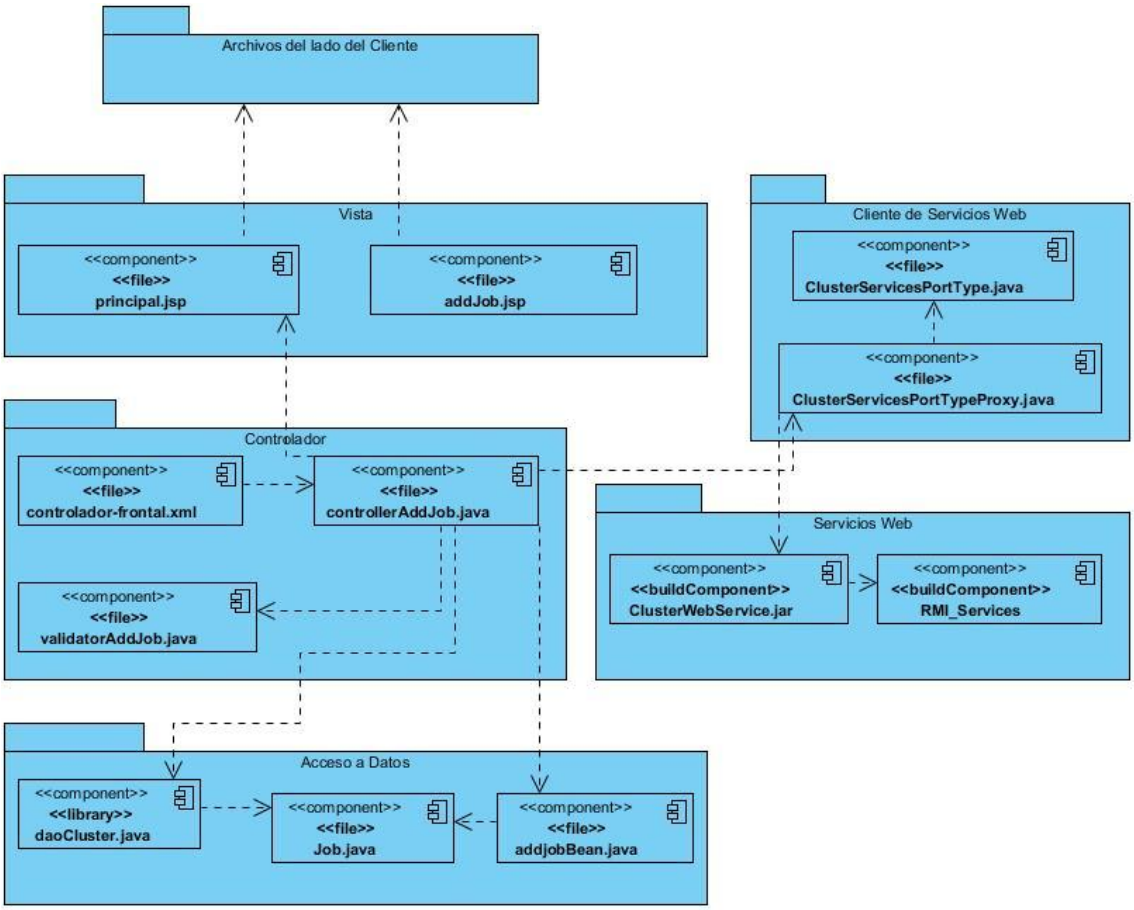


Figura 8. Diagrama de Componentes del CU "Gestionar Tareas" de la sección "Añadir Tarea".

Un componente es una parte física y reemplazable de un sistema conformado con un conjunto de interfaces y proporciona la realización de dicho conjunto. Se usan para modelar los elementos que pueden hallarse en un nodo por lo que empaquetan elementos como clases e interfaces. Los componentes mostrados en el diagrama de la Figura 8 se resumen en la siguiente tabla (Tabla 3) que mostramos a continuación.

Tabla 33. Descripción de los componentes del CU "Gestionar Tareas". En la tabla se describen los componentes que interactúan con el CU "Gestionar Tareas".

Componente	Propósito del componente.
Archivos del lado del Cliente.	Contiene los estilos CSS y los archivos Java Script que se ejecutan en el navegador del cliente.
principal.jsp	Página jsp donde se mostrará la lista de tareas al usuario y donde podrá acceder a opciones como por ejemplo "Añadir una nueva tarea".
addJob.jsp	Página jsp mediante la cual el usuario introducirá los datos que completarán la creación de una nueva tarea.
controlador-frontal.xml	Archivo de configuración encargado de atender todas las peticiones y enviarlas a las clases controladoras encargadas.
controllerAddJob.java	Clase java encargada de gestionar todas las acciones correspondientes a adicionar una nueva tarea.
validatorAddJob.java	Clase java encargada de las validaciones de los datos que introduzca el usuario a la hora de añadir una nueva tarea.
daoCluster.jar	Librería que da acceso al servicio web a las funciones del clúster.
Job.java	Clases java la cual representa la estructura de una tarea que podemos enviar al clúster.
addJobBean.java	Clase java que representa una tarea y contiene los parámetros necesarios para añadir esta al clúster.
ClClusterServicesPortTypeProxy.java	Clase que implementa las operaciones de lainterfazClusterServicesPortType.java.
ClusterServicesPortType.java	Clase interfaz que declara todas las operaciones que invocan los servicios web.
ClusterWebServices.jar	Programa ejecutable que consume los servicios web implementados.

RMI_Services	Aplicación que consume los servicios implementados directamente del clúster.

4.1.2 Modelo de despliegue.

El diagrama de despliegue es un diagrama en forma de grafo donde sus nodos pueden ser componentes de software y/o objetos. Cada nodo representa una unidad computacional de algún tipo, por ejemplo un dispositivo. Estos componentes suelen estar unidos por relaciones de dependencia las cuales pueden ser múltiples ya que un componente puede contar con más de una interfaz de acceso o comunicación.

A continuación mostramos el modelo de despliegue del sistema (Figura 9). Las conexiones entre los nodos son en ambas direcciones. Esta vista permite determinar las consecuencias de la distribución y la asignación de los recursos. Las instancias de los nodos pueden contener instancias de ejecución, de componentes y objetos.

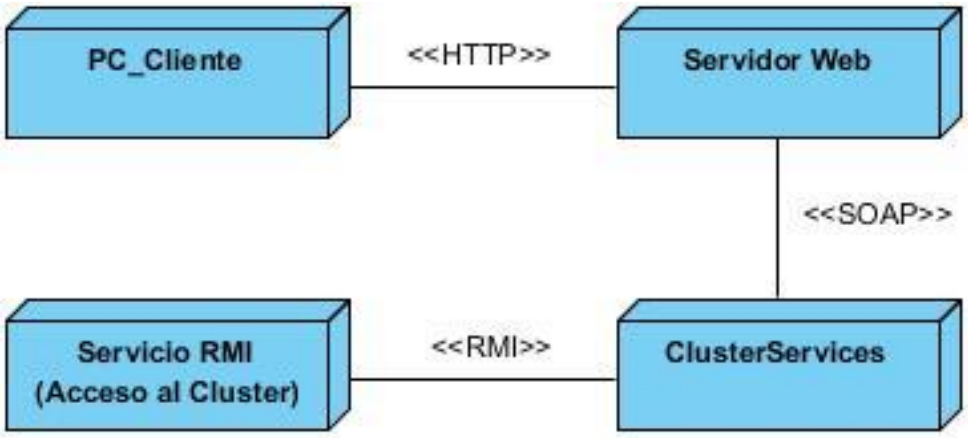


Figura 9. Diagrama de Despliegue.

Para el correcto despliegue del software se necesita al menos de una computadora (PC) con un navegador web, que permita realizar peticiones a través del protocolo de red HTTP al servidor de aplicaciones, el cual encapsula la lógica de aplicación. Por último las funcionalidades están accesibles en el servidor de servicio a través de SOAP.

4.2 Pruebas de Software

Cuando un sistema tiene terminado una serie de clases y componentes, es recomendable que el software sea probado para encontrar en caso de que existan y corregir el máximo de errores posible. El objetivo es diseñar una serie de casos de prueba que tengan una alta probabilidad de encontrar errores. Para ello existen técnicas de prueba de software. Estas técnicas facilitan una guía sistemática para diseñar pruebas que: comprueben la lógica interna de los componentes del software y verifiquen los dominios de entrada y salida del programa para descubrir errores en la funcionalidad, el comportamiento y rendimiento. Para ser más eficaces, las pruebas deberían ser realizadas por un equipo externo a los desarrolladores, esto garantiza pruebas con alta probabilidad de encontrar errores, que es el objetivo principal de las pruebas. Existen dos tipos de pruebas, las de caja blanca y las de caja negra (35).

4.2.1 Pruebas de caja blanca

Las pruebas de caja blanca, se aplica mediante un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para obtener los casos de prueba que serán aplicados. Mediante los métodos de prueba de caja blanca se pueden obtener casos de prueba que garanticen que: se ejercite por lo menos una vez todos los caminos independientes de cada módulo; ejerciten todas las decisiones lógicas en sus vertientes verdaderas y falsas; ejecuten todos los bucles en sus límites y con sus límites operacionales; y ejerciten las estructuras internas de datos para asegurar su validez.

Uno de los métodos más usados en las pruebas de caja blanca es el camino básico, la cual determina la complejidad ciclomática de una porción de código. La complejidad ciclomática es una métrica del software que proporciona una medición cuantitativa de la complejidad lógica de un programa (35).

Cuando se usa el camino básico, el valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y nos da un límite superior para el número de pruebas que se deben realizar. Esta complejidad se puede calcular de tres formas:

1. El número de regiones del grafo de flujo coincide con la complejidad ciclomática.
2. La complejidad ciclomática, V(G), de un grafo de flujo G, se define como: V(G)=A-N+2.Donde A es el número de aristas del grafo de flujo y N es el número de nodos del mismo.

3. La complejidad ciclomática, V(G), de un grafo de flujo G también se define como V(G)= P +1. Donde P es el número de nodos predicado[5] contenidos en el grafo de flujo G.

4.2.2 Pruebas de caja negra

Las pruebas de caja negra o pruebas de comportamientos, chequean los requisitos funcionales del software. Estas pruebas no son una alternativa a las técnicas de prueba de caja blanca. Más bien, se trata de un enfoque complementario que intenta descubrir diferentes tipos de errores que los que identifican los métodos de caja blanca (35). La prueba de caja negra intenta identificar errores de las siguientes categorías: funciones incorrectas o ausentes, errores de interfaz, errores en estructuras de datos o en accesos a bases de datos externas, errores de rendimiento y errores de inicialización y de terminación. Existen varios métodos de caja negra entre los que se pueden mencionar los métodos de prueba basados en grafo, partición equivalente y análisis de valores al límite.

La partición equivalente es un método de prueba de caja negra que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. Un caso de prueba ideal descubre de con facilidad errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico (35). La partición equivalente se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar.

Las clases de equivalencia se pueden definir de acuerdo con las siguientes directrices:

1. Si una condición de entrada especifica un rango, se define una clase de equivalencia válida y dos no válidas.
2. Si una condición de entrada requiere un valor específico, se define una clase de equivalencia válida y dos no válidas.
3. Si una condición de entrada especifica un miembro de un conjunto, se define una clase de equivalencia válida y una no válida.
4. Si una condición de entrada es lógica, se define una clase de equivalencia válida y una no válida.

4.3 Aplicación de pruebas de caja blanca

Uno de los métodos de prueba de caja blanca es la del camino simple, que se aplica a fragmentos de código de la aplicación. Se determinó aplicar este método a la función encargada de listar los usuarios

[5] Nodos a partir de los que se puede tomar más de un camino.

existentes en el clúster. A continuación en la Figura 10 el método al cual se le aplicó las pruebas de caja blanca y en la Figura 11 el camino básico resultante de aplicar las pruebas al método.

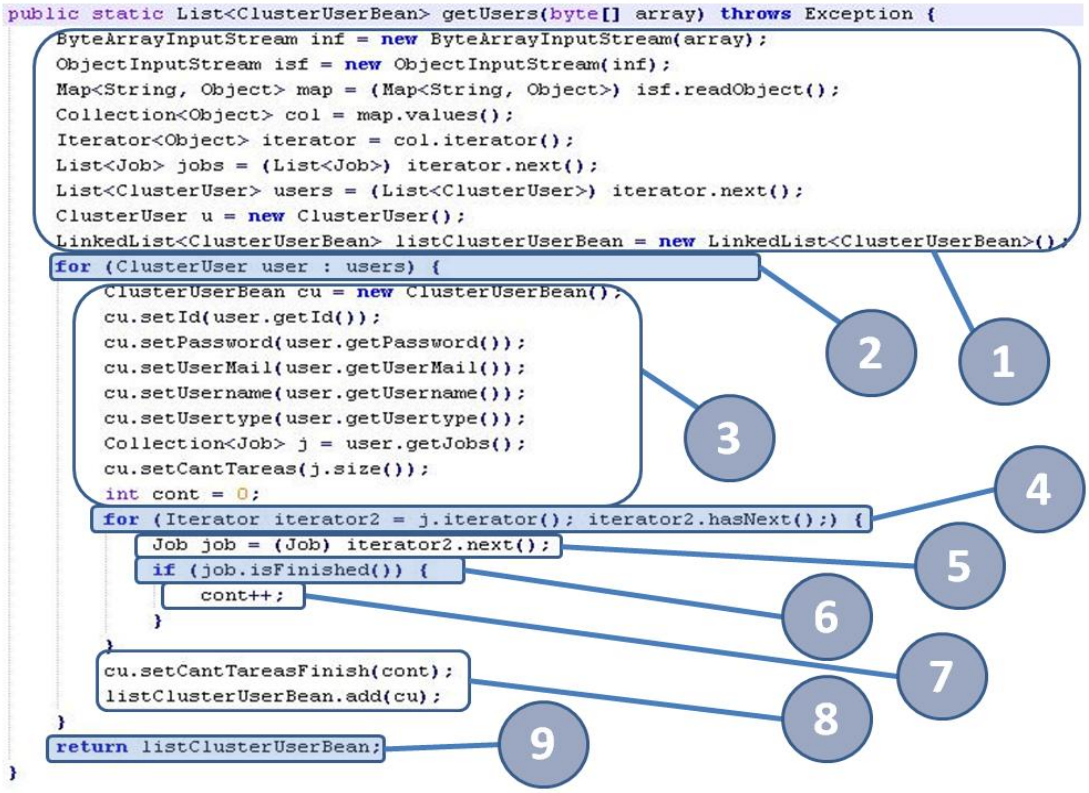


Figura 10. Función encargada de convertir los datos recibidos en una lista de Usuarios.

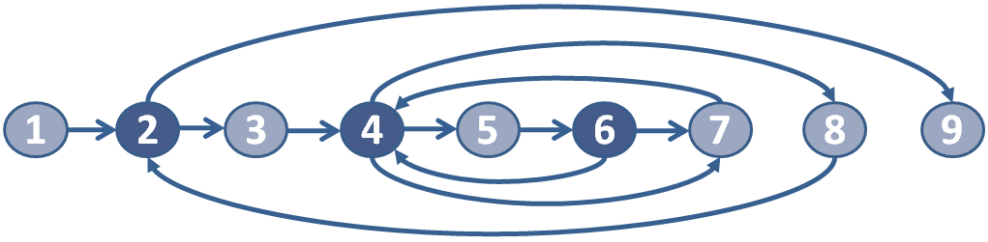


Figura 11. La gráfica muestra el camino básico para la función. Los nodos resaltados muestran los nodos predicados los cuales corresponden a las condiciones de la función. Las aristas indican los posibles caminos a recorrer en cada uno de los casos.

Con el camino básico determinado, se aplica una de las tres formas para calcular la complejidad ciclomática, se utilizó la fórmula V(G) = A-N + 2, para la cual se obtuvo once artistas y nueve nodos, por lo tanto: V(G) = 12-9 + 2, quedando V(G) = 5. La complejidad ciclomática indica los posibles casos de ejecución para la función, lo que sería de utilidad a la hora de diseñar los casos de prueba de caja negra.

4.4 Casos de pruebas de caja negra

Antes de comenzar a realizar los casos de prueba es necesario definir las variables que serán utilizadas para su desarrollo.

Para la sección “Adicionar Tarea” del CU “Gestionar Tareas” se definen las siguientes variables:

Tabla 4. Variables para el caso de prueba “Adicionar Tarea”.

No.	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Comandos	Campo de texto	No	Se especifica el nombre de la aplicación que se quiere ejecutar.
2	Parámetros	Campo de texto	No	Se especifica la dirección del fichero donde se encuentran los parámetros de la tarea.

Tabla 5. Caso de prueba 1: “Adicionar Tarea”.

Escenario	Descripción	Comandos	Parámetros	Respuesta del Sistema	Flujo central
EC 1.1El Investigador selecciona añadir una nueva tarea e introduce los datos de la misma.	El Investigador selecciona en el menú principal la opción añadir nueva tarea. El sistema muestra un formulario donde el Investigador puede introducir los datos de la nueva tarea.	V	V	El sistema muestra la interfaz principal con una lista de las tareas existentes incluyendo la nueva tarea adicionada.	1- Añadir Nueva Tarea 2- Llenar los datos 3- Añadir
EC 1.2 El investigador introduce datos	El Investigador al adicionar la nueva	I	V	El sistema muestra un aviso que se	1- Añadir Nueva

erróneos o deja campos en blanco.	tarea deja campos obligatorios del formulario sin llenar o los llena incorrectamente.	V	I	dejaron campos obligatorios sin llenar y espera que el Investigador realice una acción.	Tarea 2- Llenar los datos 3- Añadir
		I	I		

Para la sección “Adicionar Usuario” del CU “Gestionar Usuarios” se definen las siguientes variables:

Tabla 6. Variables para el caso de prueba “Adicionar Usuario”.

No.	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Usuario	Campo de texto	No	Se especifica un nuevo nombre de usuario para añadir.
2	Contraseña	Campo de texto	No	Se especifica la contraseña para el nuevo usuario que se va a añadir.
3	Repita la Contraseña	Campo de texto	No	Se repite la contraseña para validar que se introdujo correctamente.
4	E-mail	Campo de texto	No	Se introduce la dirección de correo electrónico del nuevo usuario para hacerle llegar las notificaciones mediante su buzón de correo.

Tabla 7. Caso de prueba 2: "Adicionar Usuario".

Escenario	Descripción	Usuario	Contraseña	Repita la Contraseña	E-mail	Respuesta del Sistema	Flujo central
EC 2.1El Investigador selecciona añadir un nuevo usuario e introduce los datos del mismo.	El Investigador selecciona en el menú principal la opción "Gestionar Usuarios" luego allí selecciona añadir nuevo usuario. El sistema muestra un formulario donde el Administrador puede introducir los datos del nuevo usuario.	V	V	V	V	El sistema muestra la interfaz "Gestionar Usuarios" con una lista de los usuarios existentes incluyendo el nuevo usuario adicionado.	1- Gestionar Usuarios 2- Añadir Usuario 3- Llenar los datos 4- Añadir
EC 2.2 El investigador introduce datos erróneos o deja campos en blanco.	El Investigador al adicionar el nuevo usuario deja campos obligatorios del formulario sin llenar o los llena incorrectamente.	V	I	I	I	El sistema muestra un aviso que se dejaron campos obligatorios sin llenar y espera que el Investigador realice una acción.	1- Gestionar Usuarios 2- Añadir Usuario 3- Llenar los datos 4- Añadir
		I	V	I	I		
		I	I	V	I		
		I	I	I	V		

Tabla 8. Variables para el caso de prueba "Editar Usuario".

No.	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Contraseña	Campo de texto	No	Se especifica la contraseña para el nuevo usuario que se va a añadir.
2	Repita su Contraseña	Campo de texto	No	Se repite la contraseña para validar que se introdujo correctamente.

3	E-mail	Campo de texto	No	Se introduce la dirección de correo electrónico del nuevo usuario para hacerle llegar las notificaciones mediante su buzón de correo.

Tabla 9. Caso de prueba 3: "Editar Usuario".

Escenario	Descripción	Contraseña	Repita la Contraseña	E-mail	Respuesta del Sistema	Flujo central
EC 2.1El Investigador selecciona editar un usuario y modifica los datos del mismo.	El Investigador selecciona en el menú principal la opción "Gestionar Usuarios" luego allí selecciona Editar un usuario. El sistema muestra un formulario donde el Administrador puede introducir los nuevos datos del usuario.	V	V	V	El sistema muestra la interfaz "Gestionar Usuarios" con una lista de los usuarios existentes incluyendo las modificaciones realizadas.	1- Gestionar Usuarios 2- Editar Usuario 3- Llenar los datos 4- Añadir
EC 2.2 El investigador introduce datos erróneos o deja campos en blanco.	El Investigador al editar el usuario deja campos obligatorios del formulario sin llenar o los llena incorrectamente.	V	I	I	El sistema muestra un aviso que se dejaron campos obligatorios sin llenar y espera que el Investigador realice una acción.	1- Gestionar Usuarios 2- Editar Usuario 3- Llenar los datos 4- Añadir
		I	V	I		
		I	I	V		

La aplicación de los casos de prueba permitió verificar el cumplimiento de los requisitos funcionales del sistema, de igual forma corroboró la robustez del sistema en el manejo de errores y situaciones anormales.

4.5 Conclusiones

En este capítulo se construyó el modelo de implementación correspondiente, para ello se realizó el diagrama de componentes del CU: “Gestionar Tareas”, específicamente para la sección “Añadir Tarea”, en el que se definieron los componentes del sistema necesario para el correcto funcionamiento de este CU. Los componentes fueron separados por paquetes como se definió en el modelo del diseño del sistema y se describió cada uno de ellos para lograr una mayor comprensión acerca de su función. Además se definió el modelo de despliegue que muestra los elementos necesarios para realizar el despliegue e instalación del sistema: al menos un ordenador cliente, un servidor de aplicaciones y un servidor de servicios web. Se analizaron las principales técnicas y métodos de pruebas de software, tanto de caja blanca como de caja negra. Se determinó aplicar el método del camino básico, correspondiente al método de caja blanca, a la función encargada de convertir los datos recibidos en una lista de usuarios, la cual arrojó un valor de 5 como complejidad ciclomática de la misma. Por otro lado se definieron los casos de prueba de caja negra para la técnica de partición de equivalencia y análisis de valores al límite a los CU: “Gestionar Tareas” y “Gestionar Usuarios”, los que se aplicaron a las secciones: “Adicionar Tarea”, “Adicionar Usuario” y “Editar Usuario” respectivamente. Estas pruebas se realizaron a través de la interfaz del sistema y se detectaron inconformidades como campos no validados y errores no tratados; todas ellas solventadas rápidamente mediante la corrección de los objetos correspondientes.

CONCLUSIONES GENERALES.

Concluidas las investigaciones podemos decir que se desarrolló un portlet que permite la gestión de las tareas y usuarios del Clúster Beowulf del Departamento de Bioinformática del Centro DATEC. Para ello:

- Se diseñó un portlet el cual brinda al usuario la posibilidad de gestionar con mayor facilidad y en un ambiente más agradable las tareas y usuarios del clúster.
- Se definieron los requisitos de la aplicación resultando 10 RF agrupados en 3 casos de uso, identificando como arquitectónicamente significativo el caso de uso “Adicionar Tarea”. La arquitectura se confeccionó con la utilización de varios patrones arquitectónicos y de diseño destacándose el MVC como patrón de diseño el cual da a la aplicación una mayor flexibilidad al cambio pues se obtiene un diseño separado por capaz.
- Se implementaron los requisitos funcionales identificados obteniendo una aplicación que permite gestionar las tareas y usuarios del clúster. También el usuario puede visualizar en tiempo real el estado de sus tareas y descargarlas en caso de haber finalizado.
- Se realizaron pruebas de caja blanca a métodos críticos de la aplicación validando los posibles caminos y obteniendo con el cálculo de la complejidad ciclomática la complejidad lógica de algunos de los métodos críticos de la aplicación.
- Se validaron las interfaces del sistema a través de la aplicación de pruebas de caja negra. Las pruebas permitieron encontrar errores e inconformidades que posteriormente fueron solucionados para entregar al cliente un producto de alta calidad.

RECOMENDACIONES.

- Dado que la gestión de un clúster puede extenderse hacia muchos más puntos que los abordados en la aplicación que se obtuvo, se recomienda mejorar la misma para brindar al cliente una mayor facilidad a la hora de gestionar y monitorizar el clúster del Departamento de Bioinformática.

- La obtención de las métricas de un clúster es una de las tareas más demandadas por sus administradores, por esto se recomienda una interfaz que permita monitorizar el clúster la cual pueda ser integrada al Portal de Servicios Bioinformáticos.

REFERENCIAS BIBLIOGRÁFICAS.

1. **Fabián Ciappina, Marcelo y Rubén Cravero, Walter.** Usando herramientas del sistema de colas para acelerar y hacer escalables nuestras rutinas seriales. [En línea] 2006. [Citado el: 12 de febrero de 2013.] www.saber.ula.ve/bitstream/123456789/16032/1/ciappina.pdf.

2. Oracle Grid Engine. [En línea] [Citado el: 12 de febrero de 1013.] http://www.oracle.com/us/products/tools/oracle-grid-engine-075549.html.

3. **Altair Grid Technologies.** [En línea] 23 de April de 2005. [Citado el: 2 de marzo de 2013.] www.altair.com.

4. **Bayucan, Albeaus, y otros.** [En línea] agosto de 2000. [Citado el: 2 de marzo de 2013.] www.veridian.com.

5. Adaptive Computing. [En línea] [Citado el: 2 de marzo de 2013.] http://www.adaptivecomputing.com/products/open-source/torque.

6. **Martín Llorente, Ignacio.** *Gestión de Recursos Distribuidos.* Madrid, España : s.n., 2005.

7. Cluster Resources. [En línea] 2010. [Citado el: 3 de marzo de 2013.] http://www.clusterresources.com/torquedocs21/p.introduction.shtml.

8. *Desarrollo de aplicaciones web.* **C, Mateu.** Cataluña, España : s.n., 2004.

9. **The Apache Software Foundation.** Apache Tomcat. [En línea] 2013. [Citado el: 4 de marzo de 2013.] http://tomcat.apache.org.

10. Adaptación de Contenidos Web Considerando las Características de los Dispositivos de Acceso. [En línea] [Citado el: 4 de marzo de 2013.] http://atreides.udg.edu/bcds/images/bcds/papers/pdf/adaptaci_n_de_contenidos_web_considerando_las_caracter_sticas_ de_los_dispositivos_de_acceso.pdf.

11. Introducción a Apache Tomcat. [En línea] [Citado el: 5 de marzo de 2013.] http://www.lsi.us.es/ docencia/get.php?id=1923.

12. **de Leon Estupiñán, J.** Portales y Portlets Web. [En línea] 2011. [Citado el: 7 de marzo de 2013.] http://www.slideshare.net/ jossydeleon/portales-y-portlets-web-9377907.

13. **Díaz León, Andry Daniel.** *Módulo básico de la Plataforma de Servicios Bioinformáticos.* La Habana : s.n., 2012.

14. **Geer, D.** Eclipse becomesthedominant Java IDE. *IEEEXplorer digital library.* [En línea] [Citado el: 7 de marzo de 2013.] http://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=31455.

15. **Martínez Pérez, Orlando y Agramonte Delgado, Alina.** VIII Congreso Internacional de Informática en la salud. [En línea] 2011. [Citado el: 10 de marzo de 2013.] http://www.informaticahabana.cu/node/2505.

16. Lenguajes de Programación. [En línea] [Citado el: 13 de mazo de 2013.] http://www.lenguajes-de-programacion.com/lenguajes-de-programacion.shtml.

17. Definición de Lenguake de Programación. [En línea] [Citado el: 15 de marzo de 2013.] http://www.definicion.org/lenguaje-de-programacion.

18. IDE. [En línea] octubre de 2006. [Citado el: 20 de marzo de 2013.] http://tecnologia.glosario.net/terminos-tecnicos-internet/ide-860.html.

19. **Geer, D.** Eclipse becomesthedominant Java IDE. [En línea] [Citado el: 20 de marzo de 2013.] http://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=31455.

20. PlataformaEclipse. [En línea] [Citado el: 21 de marzo de 2013.] http://plataformaeclipse.com/faq/.

21. Glosario Digital. [En línea] 2013. [Citado el: 13 de marzo de 2013.] http://www.glosariodigital.com/termino/framework/.

22. SPRING FRAMEWORK. [En línea] http://www.springsource.org/spring-framework.

23. **Perera, S.** Middleware forNextGeneration Web Services. [En línea] [Citado el: 12 de marzo de 2013.] http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber= 4032100&abstractAccess=no&userType=inst.

24. SELECTING A DEVELOPMENT APPROACH. [En línea] marzo de 2008. [Citado el: 13 de marzo de 2013.] https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SelectingDevelopmentApproach.pdf.

25. [En línea] [Citado el: 21 de marzo de 2013.] http://www.antares.itmorelia.edu.mx/~jcolivar/courses/pm10a/pm_u3.ppt.

26. **Torres Flores, Lizet.** Establecimiento de una Metodología de Desarrollo de Software para la Universidad de Navojoa Usando OpenUP. [En línea] [Citado el: 22 de marzo de 2013.] http://fit.um.edu.mx/CIDET/publicaciones/COMP-004-2008% 20Establecimiento%20de%20una% 20Metodolog%C3%ADa%20de%20Desarrollo%20de% 20Software%20para%20la%20Universidad%20de%20Navojoa%20Usando%20OpenUP.pdf.

27. **Cornejo.** ¿Quées UML? [En línea] [Citado el: 21 de marzo de 2013.] http://www.docirs.cl/uml.htm.

28. Herramientas CASE. [En línea] [Citado el: 22 de marzo de 2013.] http://www.cyta.com.ar/biblioteca/bddoc/bdlibros/proyectoinformatico/libro/c5/c5.htm.

29. **Zapata, MC.** Diagramador una herramienta upper CASE para la obtención de diagramas UML desde esquemas preconceptuales. [En línea] 2007. [Citado el: 25 de marzo de 2013.] http://www.doaj.org/doaj?func=abstract&id=337842.

30. **Callejas Cuervo, M.** Herramientas libres para modelar software. [En línea] diciembre de 2005. [Citado el: 25 de marzo de 2013.] http://ervisual.hostoi.com/pdf/er_tecnico.pdf.

31. Agile Modeling. [En línea] [Citado el: 26 de marzo de 2013.] http://www.agilemodeling.com/artifacts/.

32. **Camacho, Erika, Cardeso, Fabio y Nuñez, Gabriel.** [En línea] abril de 2004. [Citado el: 26 de marzo de 2013.] http://prof.usb.ve/lmendoza/Documentos/PS-6116/Guia%20Arquitectura%20v.2.pdf.

33. **Bass, L, Clements, P y Kazman, R.** Software Architecture in Practice. [En línea] abril de 2003. [Citado el: 29 de marzo de 2013.]

34. **Prentice, Hall.** UML y Patrones. Introducción al análisis y diseño orientado a objetos. [En línea] [Citado el: 29 de marzo de 2013.]

35. **Pressman, Roger S.** Ingeniería del software. Un enfoque práctico. [En línea] 1998. [Citado el: 2 de abril de 2013.]

BIBLIOGRAFÍA

1. **Fabián Ciappina, Marcelo y Rubén Cravero, Walter.** Usando herramientas del sistema de colas para acelerar y hacer escalables nuestras rutinas seriales. [En línea] 2006. [Citado el: 12 de febrero de 2013.] www.saber.ula.ve/bitstream/123456789/16032/1/ciappina.pdf.

2. Oracle Grid Engine. [En línea] [Citado el: 12 de febrero de 1013.] http://www.oracle.com/us/products/tools/oracle-grid-engine-075549.html.

3. **Altair Grid Technologies.** [En línea] 23 de April de 2005. [Citado el: 2 de marzo de 2013.] www.altair.com.

4. **Bayucan, Albeaus, y otros.** [En línea] agosto de 2000. [Citado el: 2 de marzo de 2013.] www.veridian.com.

5. Adaptive Computing. [En línea] [Citado el: 2 de marzo de 2013.] http://www.adaptivecomputing.com/products/open-source/torque.

6. **Martín Llorente, Ignacio.** *Gestión de Recursos Distribuidos.* Madrid, España : s.n., 2005.

7. Cluster Resources. [En línea] 2010. [Citado el: 3 de marzo de 2013.] http://www.clusterresources.com/torquedocs21/p.introduction.shtml.

8. *Desarrollo de aplicaciones web.* **C, Mateu.** Cataluña, España : s.n., 2004.

9. **The Apache Software Foundation.** Apache Tomcat. [En línea] 2013. [Citado el: 4 de marzo de 2013.] http://tomcat.apache.org.

10. Adaptación de Contenidos Web Considerando las Características de los Dispositivos de Acceso. [En línea] [Citado el: 4 de marzo de 2013.] http://atreides.udg.edu/bcds/images/bcds/papers/pdf/adaptaci_n_de_contenidos_web_considerando_las_caracter_sticas_ de_los_dispositivos_de_acceso.pdf.

11. Introducción a Apache Tomcat. [En línea] [Citado el: 5 de marzo de 2013.] http://www.lsi.us.es/ docencia/get.php?id=1923.

12. **de Leon Estupiñán, J.** Portales y Portlets Web. [En línea] 2011. [Citado el: 7 de marzo de 2013.] http://www.slideshare.net/ jossydeleon/portales-y-portlets-web-9377907.

13. **Díaz León, Andry Daniel.** *Módulo básico de la Plataforma de Servicios Bioinformáticos.* La Habana : s.n., 2012.

14. **Geer, D.** Eclipse becomesthedominant Java IDE. *IEEEXplorer digital library.* [En línea] [Citado el: 7 de marzo de 2013.] http://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=31455.

15. **Martínez Pérez, Orlando y Agramonte Delgado, Alina.** VIII Congreso Internacional de Informática en la salud. [En línea] 2011. [Citado el: 10 de marzo de 2013.] http://www.informaticahabana.cu/node/2505.

16. Lenguajes de Programación. [En línea] [Citado el: 13 de mazo de 2013.] http://www.lenguajes-de-programacion.com/lenguajes-de-programacion.shtml.

17. Definición de Lenguake de Programación. [En línea] [Citado el: 15 de marzo de 2013.] http://www.definicion.org/lenguaje-de-programacion.

18. IDE. [En línea] octubre de 2006. [Citado el: 20 de marzo de 2013.] http://tecnologia.glosario.net/terminos-tecnicos-internet/ide-860.html.

19. **Geer, D.** Eclipse becomesthedominant Java IDE. [En línea] [Citado el: 20 de marzo de 2013.] http://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=31455.

20. PlataformaEclipse. [En línea] [Citado el: 21 de marzo de 2013.] http://plataformaeclipse.com/faq/.

21. Glosario Digital. [En línea] 2013. [Citado el: 13 de marzo de 2013.] http://www.glosariodigital.com/termino/framework/.

22. SPRING FRAMEWORK. [En línea] http://www.springsource.org/spring-framework.

23. **Perera, S.** Middleware forNextGeneration Web Services. [En línea] [Citado el: 12 de marzo de 2013.] http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber= 4032100&abstractAccess=no&userType=inst.

24. SELECTING A DEVELOPMENT APPROACH. [En línea] marzo de 2008. [Citado el: 13 de marzo de 2013.] https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SelectingDevelopmentApproach.pdf.

25. [En línea] [Citado el: 21 de marzo de 2013.] http://www.antares.itmorelia.edu.mx/~jcolivar/courses/pm10a/pm_u3.ppt.

26. **Torres Flores, Lizet.** Establecimiento de una Metodología de Desarrollo de Software para la Universidad de Navojoa Usando OpenUP. [En línea] [Citado el: 22 de marzo de 2013.] http://fit.um.edu.mx/CIDET/publicaciones/COMP-004-2008% 20Establecimiento%20de%20una% 20Metodolog%C3%ADa%20de%20Desarrollo%20de% 20Software%20para%20la%20Universidad%20de%20Navojoa%20Usando%20OpenUP.pdf.

27. **Cornejo.** ¿Quées UML? [En línea] [Citado el: 21 de marzo de 2013.] http://www.docirs.cl/uml.htm.

28. Herramientas CASE. [En línea] [Citado el: 22 de marzo de 2013.] http://www.cyta.com.ar/biblioteca/bddoc/bdlibros/proyectoinformatico/libro/c5/c5.htm.

29. **Zapata, MC.** Diagramador una herramienta upper CASE para la obtención de diagramas UML desde esquemas preconceptuales. [En línea] 2007. [Citado el: 25 de marzo de 2013.] http://www.doaj.org/doaj?func=abstract&id=337842.

30. **Callejas Cuervo, M.** Herramientas libres para modelar software. [En línea] diciembre de 2005. [Citado el: 25 de marzo de 2013.] http://ervisual.hostoi.com/pdf/er_tecnico.pdf.

31. Agile Modeling. [En línea] [Citado el: 26 de marzo de 2013.] http://www.agilemodeling.com/artifacts/.

32. **Camacho, Erika, Cardeso, Fabio y Nuñez, Gabriel.** [En línea] abril de 2004. [Citado el: 26 de marzo de 2013.] http://prof.usb.ve/lmendoza/Documentos/PS-6116/Guia%20Arquitectura%20v.2.pdf.

33. **Bass, L, Clements, P y Kazman, R.** Software Architecture in Practice. [En línea] abril de 2003. [Citado el: 29 de marzo de 2013.]

34. **Prentice, Hall.** UML y Patrones. Introducción al análisis y diseño orientado a objetos. [En línea] [Citado el: 29 de marzo de 2013.]

35. **Pressman, Roger S.** Ingeniería del software. Un enfoque práctico. [En línea] 1998. [Citado el: 2 de abril de 2013.]

ANEXOS.

Descripción del caso de uso Autenticar Usuario

Tabla 10. Descripción del CU "Autenticar Usuario".

Caso de Uso	Autenticar
Actor	Investigador.
Propósito	El investigador hace uso de sus credenciales para acceder al sistema
Resumen	El caso de uso se inicia cuando el investigador selecciona ejecutar la aplicación. Por último, se finaliza el caso de uso cuando el sistema comprueba la valides de las credenciales del investigador.
Referencia	RF1.
Precondición	Debe seleccionar ejecutar la aplicación
Prioridad	Media.
Flujo Normal de Eventos	
Acción del Autor	**Respuesta del Sistema**
1. El caso de uso se inicia cuando el investigador selecciona ejecutar la aplicación	2. El sistema muestra una interfaz para que el investigador introduzca sus credenciales.

Sección 1 “Autenticar”	
Acción del Actor	**Respuesta del Sistema**
1.1 El investigador selecciona ejecutar la aplicación.	1.2 El sistema muestra una interfaz en la cual el investigador puede introducir sus credenciales.
1.3 El investigador selecciona la opción Autenticarse.	1.4 El sistema comprueba las credenciales del investigador y procede a mostrar la interfaz principal de la aplicación.
Flujos Alternos de la sección 1 “Autenticar” del paso 1.4	
Acción del Actor	**Respuesta del Sistema**
	1.4.1. Si las credenciales del investigador no son válidas el sistema muestra un mensaje donde advierte al investigador de lo ocurrido y retorna al paso 1.2.
Poscondiciones	El Investigador debe encontrarse en la interfaz principal del sistema.

Descripción del CU “Gestionar Usuarios”

Tabla 11. Descripción del CU “Gestionar Usuarios”. En la siguiente tabla se realiza la descripción de todas las secciones del CU “Gestionar Usuarios”.

Caso de Uso	Gestionar Usuarios.
Actor	Investigador.

Propósito	Crear y modificar usuarios en el clúster.
Resumen	El caso de uso se inicia cuando el investigador va a realizar alguna de las siguientes operaciones relacionadas con la gestión de usuarios: ➢ Adicionar un nuevo usuario. ➢ Listar usuarios existentes en el clúster. ➢ Editar usuarios existentes en el clúster ➢ Eliminar usuarios existentes en el clúster. Por último, se finaliza el caso de uso cuando el sistema actualiza sus datos.
Referencia	RF8, RF9, RF10, RF11.
Precondición	Debe estar autenticado.
Prioridad	Crítico.

Flujo Normal de Eventos	
Acción del Autor	**Respuesta del Sistema**
1. El caso de uso se inicia cuando el investigador selecciona una de las siguientes opciones: ➢ Adicionar un nuevo usuario. ➢ Listar usuarios existentes en el clúster. ➢ Editar usuarios existentes en el clúster. ➢ Eliminar usuarios existentes en el clúster.	2. El sistema en dependencia de la opción seleccionada realizará lo siguiente: ➢ Si seleccionó Adicionar un nuevo usuario, ir a sección “Adicionar un nuevo usuario”. ➢ Si seleccionó Listar usuarios existentes en el clúster ir a sección “Listar usuarios existentes en el clúster”. ➢ Si seleccionó Editar usuarios existentes en el clúster ir a sección “Editar usuarios existentes en el clúster”. ➢ Si seleccionó Eliminar usuarios existentes en el clúster ir a sección

	"Eliminar usuarios existentes en el clúster".
Sección 1 "Adicionar nuevo usuario"	
Acción del Actor	**Respuesta del Sistema**
1.1 El investigador selecciona la opción "Gestionar Usuarios" en la interfaz principal.	1.2 El sistema muestra al investigador una nueva interfaz con la lista de los usuarios que sus privilegios le permiten gestionar, y muestra el botón "Añadir Usuario".
1.3 El investigador presiona en la opción presiona el botón "Añadir Usuario".	1.4 El sistema muestra una interfaz para que el investigador introduzca los datos del nuevo usuario.
1.5 El investigador introduce los datos y presiona "Añadir".	1.6 El sistema comprueba que los datos introducidos sean correctos y adiciona el nuevo usuario, luego muestra la interfaz Gestionar Usuarios con la lista actualizada de los usuarios existentes en el clúster. Fin del caso de uso.
Flujos Alternos de la sección 1 "Adicionar Usuario" del paso 1.6	
Acción del Actor	**Respuesta del Sistema**
	1.6 Si los valores insertados por el investigador no son correctos el sistema no adiciona el usuario y muestra un mensaje advirtiendo al investigador de lo ocurrido. Finaliza el caso de uso.
Sección 2 "Listar usuarios del clúster"	
Acción del Actor	**Respuesta del Sistema**
2.1 El investigador selecciona la opción	2.2 El sistema muestra al investigador

“Gestionar Usuarios” en la interfaz principal.	una nueva interfaz con la lista de los usuarios que sus privilegios le permiten gestionar.
Sección 3 “Editar usuario del clúster”	
Acción del Actor	**Respuesta del Sistema**
3.1 El investigador selecciona la opción “Gestionar Usuarios” en la interfaz principal.	3.2 El sistema muestra al investigador una nueva interfaz con la lista de los usuarios que sus privilegios le permiten gestionar.
3.3 El investigador selecciona la opción “Editar” del usuario que este quiera editar.	3.4El sistema muestra una interfaz para que el investigador modifique los datos del usuario.
3.5 El investigador introduce los datos a modificar y presiona el botón “Modificar”.	3.6 El sistema comprueba que los datos introducidos sean correctos y actualiza la información en la interfaz Gestionar Usuarios.
Flujos Alternos de la sección 3 “Editar usuario del clúster” del paso 3.6	
Acción del Actor	**Respuesta del Sistema**
	3.6Si los valores insertados por el investigador no son correctos el sistema no modifica el usuario y muestra un mensaje advirtiendo al investigador de lo ocurrido. Finaliza el caso de uso.
Sección 4 “Eliminar usuario del clúster”	
Acción del Actor	**Respuesta del Sistema**
4.1 El investigador selecciona la opción “Gestionar Usuarios” en la interfaz principal.	4.2 El sistema muestra al investigador una nueva interfaz con la lista de los usuarios que sus privilegios le permiten gestionar.

4.3 El investigador selecciona la opción "Eliminar" del usuario que este quiera eliminar.	4.4 El sistema muestra un mensaje para que el investigador confirme la acción. Finaliza el caso de uso.
Flujos Alternos de la sección 3 "Eliminar usuario" del paso 4.3	
Acción del Actor	**Respuesta del Sistema**
4.3 el investigador selecciona "cancelar" eliminar usuario.	4.4 El sistema no elimina el usuario y muestra la interfaz Gestionar Usuarios al investigador. Fin del caso de uso.
Postcondiciones	El investigador debe encontrarse en la interfaz Gestionar Usuarios del sistema.

Diagramas

Diagramas de Secuencias

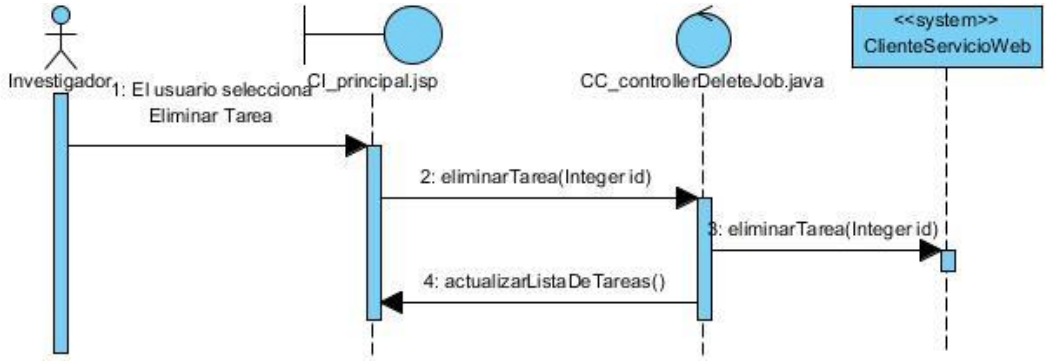


Figura 12. Diagrama de Secuencia de la sección Eliminar Tarea del CU Gestionar Tarea.

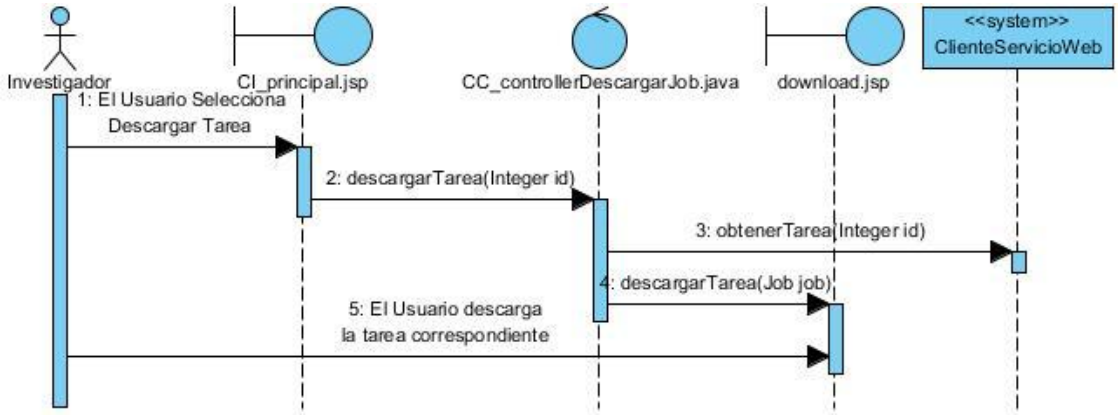


Figura 13. Diagrama de Secuencia de la sección Descargar Tarea del CU Gestionar Tarea.

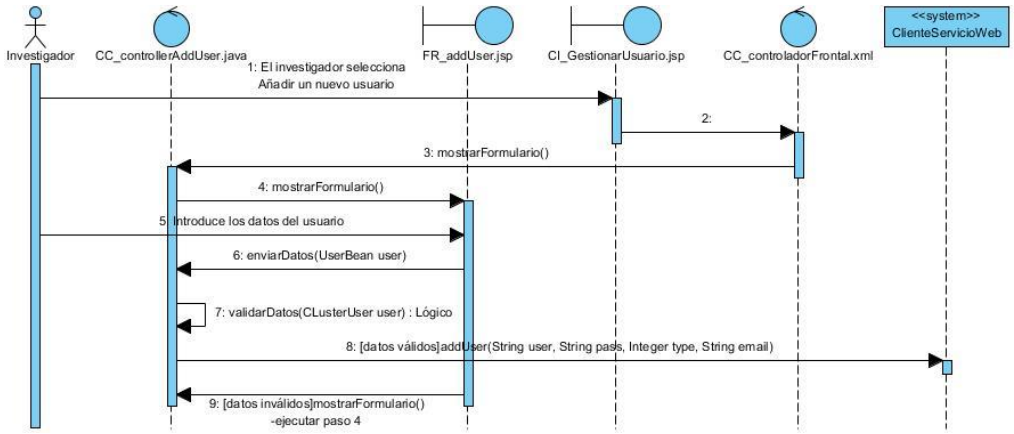


Figura 14. Diagrama de Secuencia de la sección Añadir Usuario del CU Gestionar Usuario.

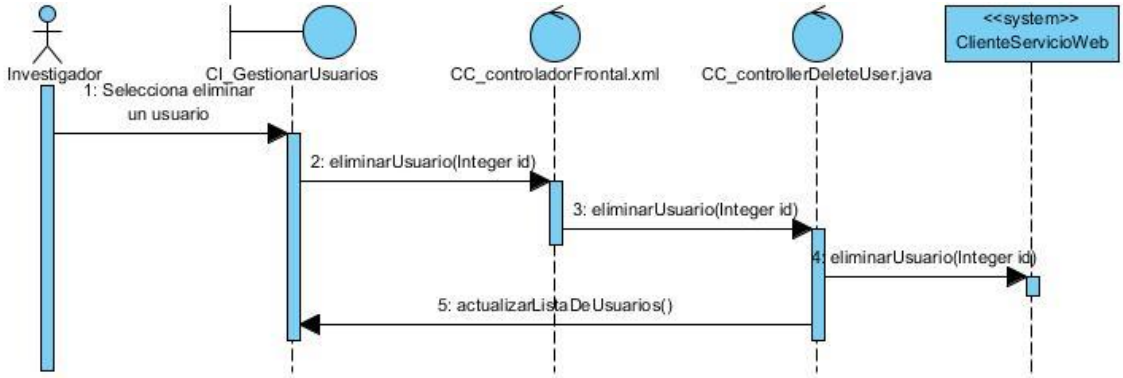


Figura 15. Diagrama de Secuencia de la sección Eliminar Usuario del CU Gestionar Usuario.

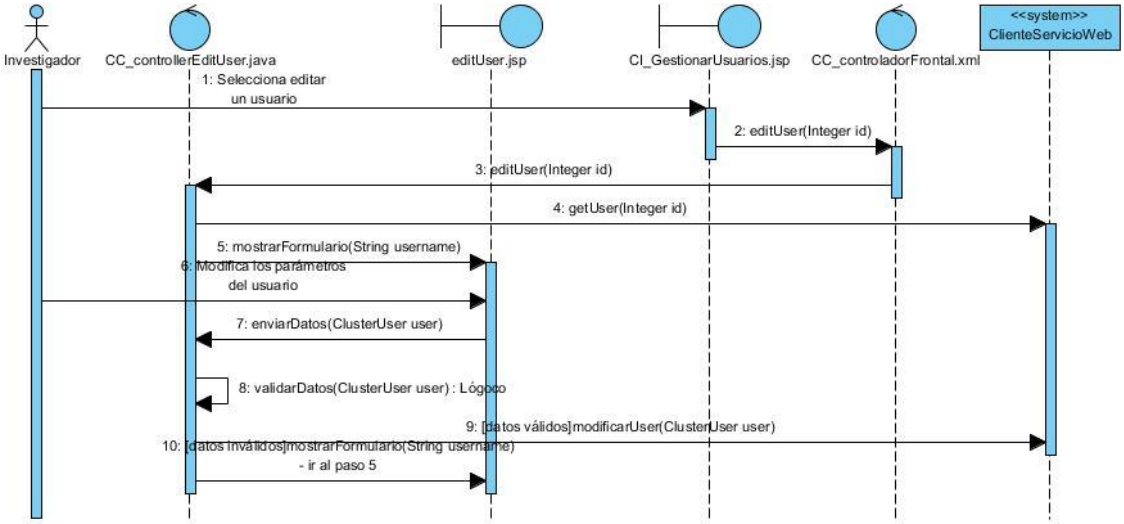


Figura 16. Diagrama de Secuencia de la sección Editar Usuario del CU Gestionar Usuario.

Diagramas de Comunicación

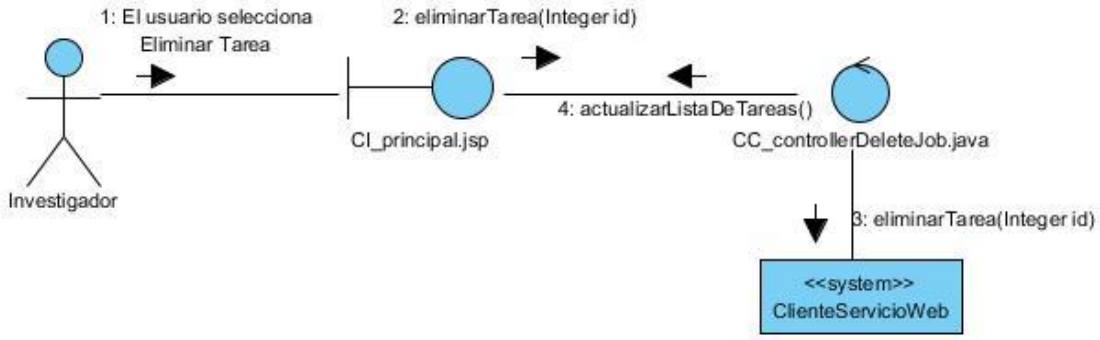


Figura 17. Diagrama de Comunicación de la sección Eliminar Tarea del CU Gestionar Tareas.

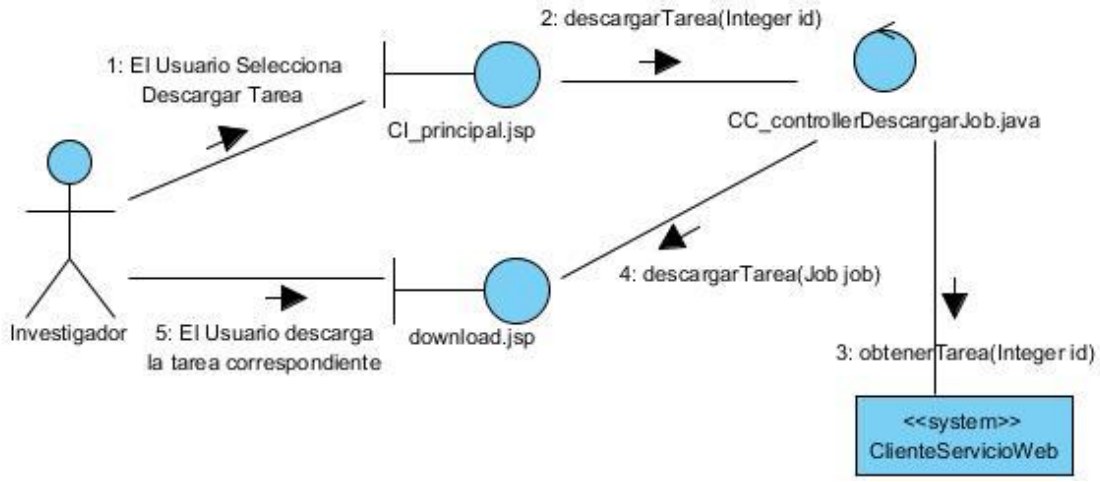


Figura 18. Diagrama de Comunicación de la sección Descargar Tarea del CU Gestionar Tareas.

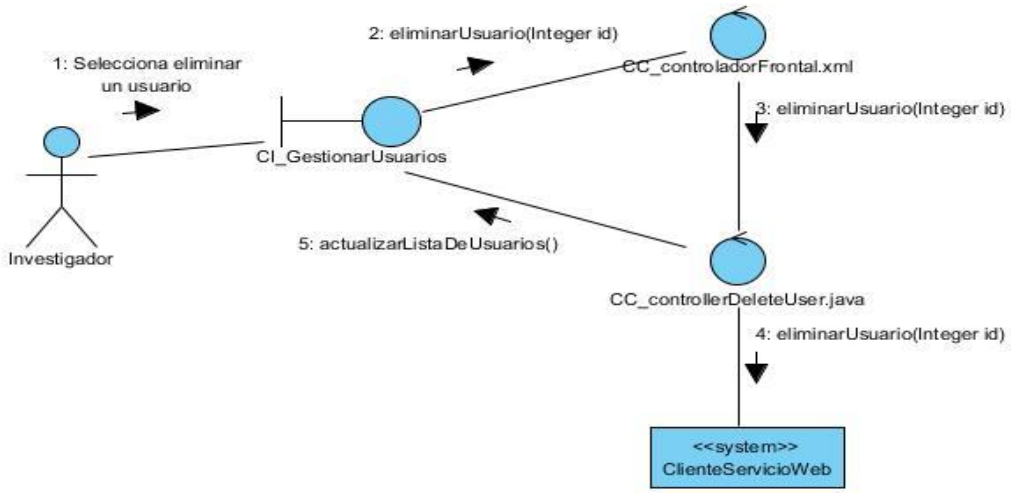


Figura 19. Diagrama de Comunicación de la sección Eliminar Usuario del CU Gestionar Usuarios.

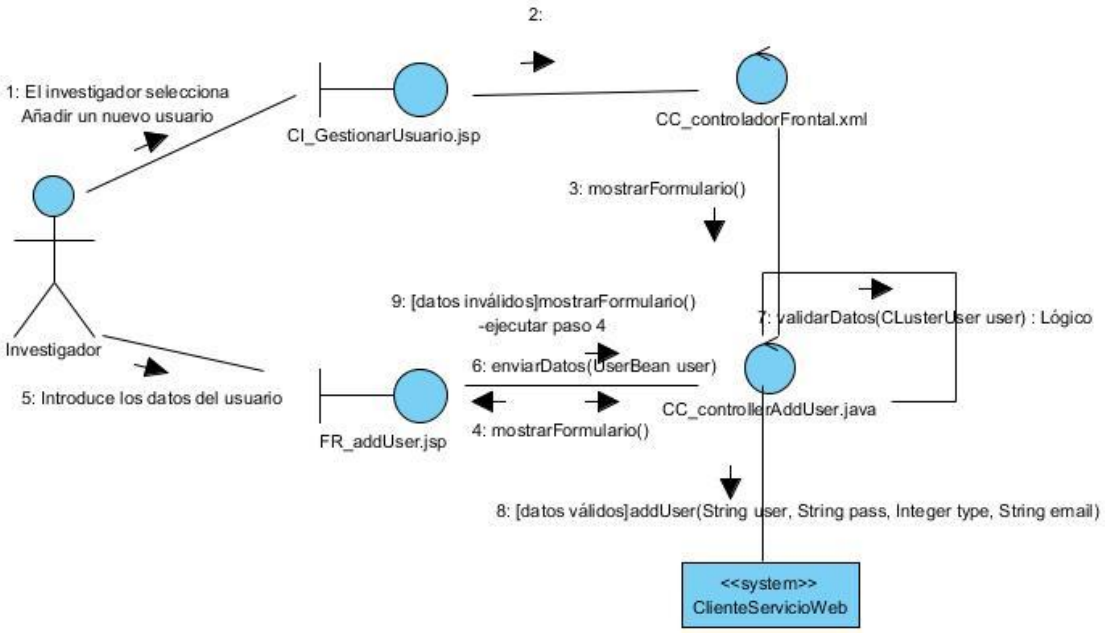


Figura 20. Diagrama de Comunicación de la sección Añadir Usuario del CU Gestionar Usuarios.

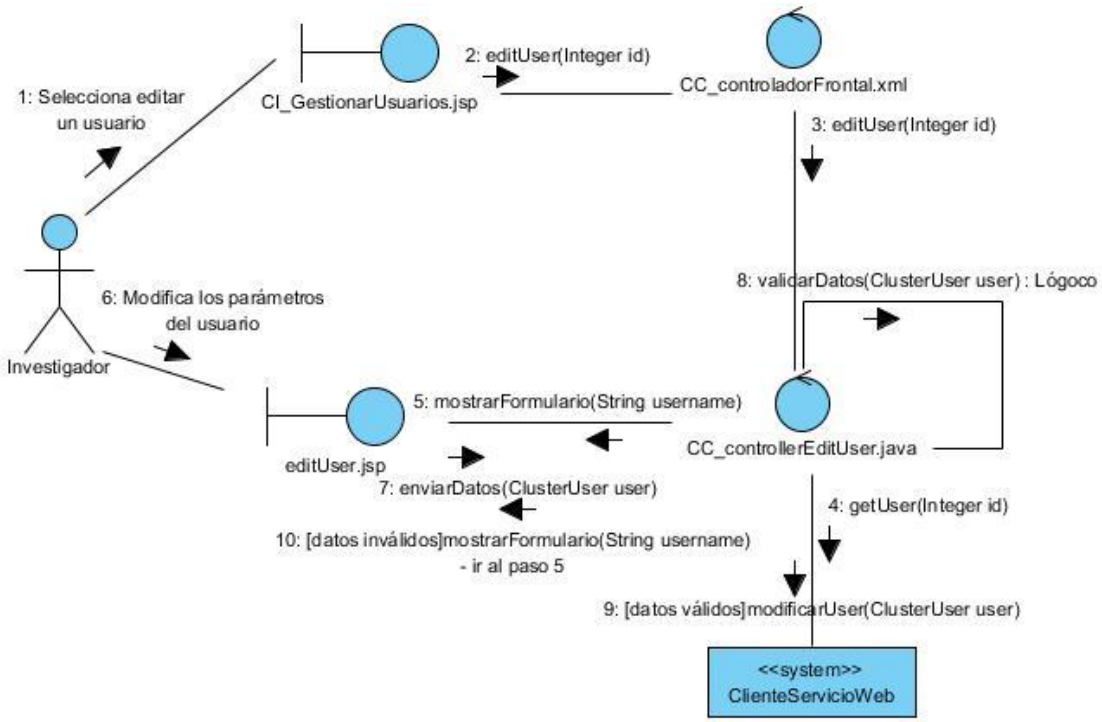


Figura 21. Diagrama de Comunicación de la sección Editar Usuario del CU Gestionar Usuarios.

GLOSARIO

Clúster Beowulf

En 1994 bajo el patrocinio del Proyecto de la Tierra y Ciencias del Espacio (ESS – Earth and SpaceSciences) del Centro de Excelencia en Datos del Espacio y Ciencias de la Información (CESDIS - Center of Excellence in Space Data and InformationSciences), Thomas Sterling y Don Becker crearon el primer clúster Beowulf con fines de investigación. Thomas Sterling nombró Beowulf al proyecto en honor al héroe de un cuento inglés, quien liberó a "Danes of Heorot" del monstruo opresivo Grendel. A manera de metáfora, se nombró Beowulf a esta nueva estrategia en computación de alto rendimiento que hace uso de tecnología bien difundida en el mercado, para derrotar a los costos opresores en tiempo y dinero de la supercomputación. Un clúster Beowulf es un grupo de computadoras compatibles con IBM, que utiliza software de código abierto. Para que un grupo de computadoras pueda ser considerado un "Beowulf" deben cumplir ciertos requisitos como, estar constituido por ordenadores personales conectados a través de redes informáticas estándar, sin el uso de equipos desarrollados específicamente para la computación paralela y contar con un monitor y un teclado.

Bioinformática

Es la ciencia encargada del procesamiento, distribución, análisis e interpretación de información biológica, mediante la aplicación de técnicas y herramientas de las matemáticas, de la biología y de la informática, con el propósito de comprender el significado biológico de una gran variedad de datos. Es el campo interdisciplinar que se encuentra en la intersección entre las Ciencias de la Vida y de la Información. Es útil para llegar a entender el flujo de información desde los genes a las estructuras moleculares, a su función bioquímica, a su conducta biológica y, finalmente, a su influencia en las enfermedades y características agronómicas. La bioinformática es en la actualidad uno de los campos de la ciencia más dinámicos y con más proyección. Es una disciplina entre la biología molecular, la informática y la estadística, que permite usar los ordenadores para investigar la biología de nuestras moléculas.