Universidad de las Ciencias Informáticas

Facultad 6

Título: Integración de los Módulos Simulación y Análisis del software alasBioSyS a la Plataforma de Servicios Bioinformáticos.

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

Autores: Yanier Scantlebury Sorí
Carlos Alberto Reinoso Cabrera
Tutores: Ing. Edel Moreno Lemus
Ing. Carlos González Iglesias

La Habana, junio 2012
Año 54 de la Revolución

"Precisamente hoy que piensas que tu vida no tiene sentido, piensa que el mayor sentido de tu vida será comprender, que mil derrotas no te hacen un ser vencido. Si luchas puedes perder, si no luchas estás perdido."

Los Aldeanos, 2008

DECLARACIÓN DE AUTORÍA

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

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

Yanier Scantlebury Sorí

_____
Firma del Autor

Carlos A. Reinoso Cabrera

_____
Firma del Autor

Edel Moreno Lemus

_____
Firma del Tutor

Carlos González Iglesias

_____
Firma del Tutor

DATOS DE CONTACTO

Ing. Edel Moreno Lemus

Ingeniero en Ciencias Informáticas, UCI 2010.

E-mail: emoreno@uci.cu

Ing. Carlos González Iglesias

Ingeniero en Ciencias Informáticas, UCI 2010.

E-mail: cgonzalez@uci.cu

AGRADECIMIENTOS

De Carlos…

Quiero agradecer primero que todo a mis padres pues son los que han hecho posible este sueño, los que han estado a mi lado en los momentos más duros y felices de mi vida. A mi papá por estar siempre dándome su apoyo, consejo y brindarme más que su cariño su amistad, por inculcarme valores como la humildad y la honradez. A mi madre por ser lo más importante en mi vida, por ser tan cariñosa, preocupada y protectora, por creer en mí siempre y decirme "yo sé que tú puedes". A mi hermana Aliet por quererme tanto y por su forma de ser. A mi abuela Caridad por estar pendiente de mí, desde que vine al mundo, por darme todo lo que ha podido, por ser su niño lindo como ella dice, por salir en mi defensa siempre cuando me castigaban o me regañaban cuando era niño y me portaba mal. A mi abuelo Rolando por luchar a brazo partido con mi abuela para que no me faltara nada, por brindarme su amor, confianza y respeto. A mi abuelo Gracielo que ya no se encuentra entre nosotros pero siempre me brindo su amor. A Celia, a Joanny, a mis tíos, a mis primos.

Quiero agradecer también a mis amigos, los del barrio que comenzaron la escuela conmigo, a Omar en especial por ser más que un amigo casi un hermano. A los que hice al entrar a la universidad, en especial a los del 6105: Adrián, Lassie, Anisleisdy, Amaia, Itamar, Mabel, la cual fue más que una amiga con la cual compartí momentos buenos y malos, a Luis Ernesto el cual es como un hermano y aunque no puede estar presente físicamente, me ha brindado ánimo y apoyo a través de estos años. A los que no pudieron terminar y se fueron por una u otra razón. A mis amigos del edificio, con los cuales he compartido momentos inolvidables, y a los cuales no quiero mencionar por temor a olvidar algún nombre. A mis compañeros del equipo de pelota, con los cuales he pasado momentos muy gratos y otros muy tristes, en especial a Ricardo. A mis compañeros del aula, del cuarto y del piquete del ocio, Ariel, Reisel, Denis, Lázaro, Jorge. A las muchachas del grupo por ser todas muy lindas y solidarias. A todos ellos por brindarme su amistad a lo largo de esta contienda, muchas gracias.

A todos los profesores que he tenido a través de esta carrera, los cuales son responsables de que haya alcanzado este resultado, algunos de ellos me brindaron más que conocimiento, educación y una simple relación alumno – profesor, me brindaron su amistad. A Milena, a Niurka, a Yanelis, a Yania, a Yunet, a Tonysé, a Yixander y a Yoamel lleguen mis más sinceros agradecimientos.

A mis tutores Edel y Carlos por guiarme en esta recta final de la carrera, por enseñarme, aconsejarme y estar siempre dispuestos a atenderme cuando lo necesité. Por ser amigos, gracias. A mi dúo de tesis Yanier por ser ejemplo de responsabilidad y preocupación, por su carácter, por ser mi amigo a pesar de mis defectos. Por haber sido el mejor compañero de tesis que alguien pudiese tener.

De Yanier…

Agradezco con infinito amor a mis padres. Mi papá Charles, el ejemplo que siempre he seguido y que aunque algunas veces no he hecho lo que me dice, la mayoría trato de cumplirlas, eres el tipo de padre que quisiera ser en el futuro. Mi mamá María del Carmen, le agradezco todo el amor y la confianza que siempre tuvo en mí, sabedora de que iba a llegar este momento. Quisiera agradecer a mis abuelos y abuelas, aunque tres no estén conmigo físicamente, los llevo siempre en mi corazón. No puedo dejar de mencionar a alguien que ha sido un padre para mí, que me acogió sin titubeos y ha cuidado de mi durante 13 años, gracias Marino por siempre.

He encontrado a una mujer maravillosa, que durante cuatro años me ha soportado en las alegrías y las penurias, y me ha regalado el bien más preciado para una persona, los hijos. Gracias Yeni, no solo por nuestra pequeña niña Dalila, sino por todo lo que has dedicado y sacrificado por mí, espero recompensarte siempre. Así como a mis suegros Emilio y Alicia, que un día irrumpí en su casa y me acogieron siempre con mucha estima.

Agradezco sinceramente a mis tías y tíos, quiero decirles que los quiero mucho y sé cuanto me han ayudado. Tuve la suerte de poder contar con tres hermanos, mi hermano Darién, aunque siempre estamos lejos, nunca dejó de pensar en ti. Mi hermana Lanny, aunque discutimos mucho por la inmadurez que siempre tuvimos te quiero mucho. Mi hermanita pequeña Maylín, eres una niña muy cariñosa y maravillosa, espero de ti que triunfes en la vida, te quiero mucho mi hermanita.

Si tuve varios hermanos, también fui afortunado por tener innumerables primos y primas, a todos va mi más sincero agradecimiento y cariño, todos han sido muy importantes para mí.

Agradezco a todos mis profesores, especialmente a Haylín, por la confianza y el cariño que siempre hemos tenido. A todos mis amigos del preuniversitario. A mi amiga y mi hermana Arelys, fuente de confesiones y comprensiones infinitas, siempre te necesitaré. Agradecer a los amigos y amigas de la Universidad, especialmente a Ariel, no lo hubiese logrado sin ti. Una persona imprescindible en este momento ha sido mi amigo Andry, te agradezco por la ayuda desinteresada que me diste. No quisiera dejar de mencionar a otras amigos, que no son menos importantes, gracias Wilfredo, Francisco, Wilber, Soto y Manuel.

De forma general agradezco a todas las personas que se han preocupado de una forma u otra por mí. A toda la familia de Yeni, a todos los de mi pueblo natal Jicotea y en Baraguá.

Agradezco a los profesores del proyecto, todos han sido generosos conmigo, especialmente a mis tutores Edel y Carlos, quienes batallaron conmigo para poder ver este día.

Por último y no menos importante, si no por lo especial que representa para mí, agradezco a mi hija Dalila, este trabajo es pensado en gran medida para ella, deseo que te sientas orgulloso de tu papá, **TE AMO** mucho mi niña.

DEDICATORIA

De Carlos…

A mi familia que siempre ha confiado en mí, especialmente a mi mamá, a mi hermana y a mi papá.

A la Revolución Cubana sin la cual nada de esto se hubiera hecho realidad, a nuestro comandante Fidel por haber creado e impulsado esta universidad.

A mi abuelo Gracielo que aunque me sea imposible su compañía, está más presente que nunca.

De Yanier…

Dedico este trabajó a la fuerza que me impulsó a seguir, mi mamá María, mi papá Charles y mi hija Dalila.

En especial a todos mis familiares y amigos.

Con amor, a mi esposa Yeni y a toda su familia.

RESUMEN

Algunas áreas del conocimiento como la Bioinformática encuentran en las Tecnologías de la Información y las Telecomunicaciones (TICs) un proceso de transformación y ampliación de los conocimientos, así como la cooperación y el intercambio mutuo. Por esta razón se han desarrollado varios proyectos entre diferentes centros científicos del país y la Universidad de las Ciencias Informáticas (UCI) para fomentar el auge de esta disciplina. Uno de estos trabajos es el software alasBioSyS, el cual es una aplicación de escritorio para la simulación y análisis de sistemas biológicos, desarrollado en Java. El mismo puede ser de gran utilidad para la comunidad científica, pero no puede ser utilizado si no se tiene instalado en la PC del investigador. Con el objetivo de fomentar el trabajo colaborativo entre los científicos dedicados al estudio de la Bioinformática se crea una aplicación para la incorporación de las funcionalidades de los Módulos Simulación y Análisis del software alasBioSyS a la Plataforma de Servicios Bioinformáticos (PSB) a través de portlets y servicios web, permitiendo así el acceso a estas funcionalidades a todos los investigadores desde cualquier navegador web.

PALABRAS CLAVE

Bioinformática, sistemas biológicos, simulación, análisis, portlet, servicio web.

Tabla de Contenidos

ÍNDICE DE FIGURAS

Índice de Tablas

INTRODUCCIÓN

El gran desarrollo tecnológico que se evidencia en la sociedad, especialmente en las ramas de la Informática y las Telecomunicaciones, muestra que se vive en una época con mayor velocidad de evolución tecnológica que cualquiera que se haya conocido. Este desarrollo acelerado ha propiciado el surgimiento de las Tecnologías de la Información y las Comunicaciones (TICs). Las mismas están inundando la vida cotidiana de las personas y a la vez ayudando a conquistar nuevos conocimientos, algunos impensados décadas atrás.

Las TICs son accesibles a diversas esferas de la vida como la economía, la cultura, la sociedad, el deporte, la salud y la educación. Ramas de la ciencia como la Biología, la Física, la Química o la Matemática hallan en las TICs la vía para un incremento de los conocimientos, así como respuestas a problemas complejos en un menor tiempo.

En las TICs se destacan las Tecnologías Web. Las mismas juegan un papel importante en la automatización de la vida de las personas. Han permitido al ser humano establecer canales de comunicación antes impensables o fuera del alcance por su alto costo. Hoy las personas gozan del correo electrónico y la mensajería instantánea como si hubieran estado presentes toda la vida[1].

Dentro de las Tecnologías Web es válido destacar los Portales Web los cuales se definen como un punto de entrada común a una colección de recursos electrónicos integrados, donde se ofrecen una serie de servicios complementarios, tales como búsqueda interna, personalización, herramientas de comunicación, servicios de información y alerta y otros servicios específicos asociados a la tipología del portal[2].

Existen diferentes tipos de portales, algunos de los que se pueden mencionar son:

- **Horizontales:** caracterizados por ser portales de carácter general orientado a todo tipo de usuario.
- **Especializados:** destinados a usuarios que buscan temas más específicos.
- **Verticales:** proveen de información y servicios a un sector en particular, con contenidos concretos y centrados en un tema.
- **Corporativos:** proveen de información de la empresa a los empleados con acceso a web públicas o de otros sectores de portales verticales.
- **Móviles:** son sitios que permiten la conexión de los usuarios tanto desde Internet como de un teléfono móvil o PDA.

Otro tipo de portal web es el utilizado por la Bioinformática, caracterizado de manera general por poseer una colección de herramientas destinadas al trabajo con sistemas biológicos. Dispone de servicios destinados a la edición y manipulación de secuencias tales como: inversión, extracción de regiones de interés, interconversión de formatos, predicción de segmentos de transmembrana y de marcos abiertos de lectura, traducción a proteína, entre otros. Además algunos brindan acceso a bases de datos remotas.

Los portales más utilizados para consultas bioinformáticas son: el Banco de Datos de ADN de Japón (DDBJ por sus siglas en inglés), el Centro Nacional para la Información Biotecnológica (NCBI, por sus siglas en inglés), el Expasy (acrónimo de Expert Protein Analysis System) y el Instituto Europeo de Bioinformática (EBI, por sus siglas en inglés).

En Cuba se cuenta con numerosos grupos de Bioinformática presentes en algunos polos científicos del país. Los más destacados son el Centro de Ingeniería Genética y Biotecnología (CIGB) de La Habana, así como el de Camagüey, el Centro de Inmunología Molecular (CIM) y el Centro Nacional de Bioinformática de Cuba (BIOINFO). Además hay pequeños grupos en la Universidad de La Habana, así como en la Universidad Central de Las Villas Marta Abreu. A pesar del impulso que el país ha hecho en el campo de la Bioinformática, no se cuenta con un portal dedicado al estudio de esta disciplina, donde puedan existir herramientas y servicios para el manejo de datos biológicos.

La Bioinformática es una nueva disciplina dentro de la Biología, donde se combinan ciencias como la Física, la Matemática y la Química; con el uso de herramientas de la computación. Utiliza las tecnologías de la información para organizar, analizar y distribuir información biológica con el propósito de responder preguntas complejas en Biología.
Si bien algunos restringen el rango de estudio de la Bioinformática al manejo y análisis de bases de datos biológicas -principalmente de secuencias-, podría atribuírsele un sentido más amplio, como la fusión de las técnicas computacionales con el entendimiento y apreciación de datos biológicos, el almacenamiento, recuperación, manipulación y correlación de datos procedentes de distintas fuentes[3].

Cuba se encuentra inmersa en un proceso de informatización de la sociedad apostando por los avances de la informática a nivel mundial. En el año 2002 fue creada la Universidad de las Ciencias Informáticas (UCI) por el Comandante en Jefe Fidel Castro con el objetivo de formar profesionales capacitados en el desarrollo de software. La UCI es una universidad privilegiada con respecto a otras instituciones del país, ya que posee amplios recursos tecnológicos y computacionales tales como

servidores con amplio procesamiento de cálculo, acceso a Internet para la descarga de programas y artículos científicos, entre otros.

Debido a la importancia estratégica que posee actualmente el desarrollo de áreas científicas como la Bioinformática se crea el Departamento de Bioinformática, perteneciente al centro DATEC de la Facultad 6. El departamento se propuso realizar un portal que brinde acceso a programas y servicios para los investigadores de universidades e instituciones del país, dedicados al estudio de la Bioinformática. Este portal será concebido como la Plataforma de Servicios Bioinformáticos y se encuentra aún en desarrollo.

Este departamento cuenta con varias líneas de producción entre las que se pueden mencionar los software siRNADesign, T-Arenal, así como el software alasBioSyS, el cual permite a los investigadores hacer uso de algoritmos y técnicas de minería de datos para realizar análisis, sustituyendo de esta manera los métodos existentes basados en la observación, los cuales son muy inexactos y representan una gran fuente de error[4]. En esta herramienta computacional se pueden realizar simulaciones a diferentes modelos matemáticos, los cuales son descritos por ecuaciones diferenciales. Además puede el investigador realizar análisis por reglas, de bifurcaciones, de estabilidad, entre otras funcionalidades.

La herramienta alasBioSyS está desarrollada con el lenguaje de programación Java en una versión de escritorio, por lo que trae consigo la instalación de la Máquina Virtual de Java en la computadora. El software utiliza también la herramienta T-Arenal, la cual debe estar disponible en un servidor para la ejecución de los problemas descritos por el usuario. Además necesita de soporte para las nuevas versiones, ocasionando el traslado de los desarrolladores de la aplicación hasta la computadora del cliente. Esto es un proceso muy engorroso debido a que genera pérdida de tiempo al cliente, así como de los desarrolladores. Por lo anteriormente planteado y a pesar de que se pudiera compartir el software en un servidor FTP nacional, permitiendo la descarga del producto; el mismo no promueve un trabajo colaborativo entre los investigadores de los diferentes institutos científicos y universitarios, en el marco de compartir experiencias con la herramienta.

La integración de la herramienta alasBioSyS a la Plataforma de Servicios Bioinformáticos permitiría la utilización de las funcionalidades de este software desde cualquier lugar a través de Internet o de la intranet nacional.

Por lo anteriormente planteado el **problema a resolver** es: la necesidad de tener acceso a las funcionalidades de los Módulos Análisis y Simulación del software alasBioSyS a través de la web.

Se plantea como **objeto de estudio** las Tecnologías Web y se define como **campo de acción** los portlets y servicios web para la Plataforma de Servicios Bioinformáticos.

Para la solución del problema anterior se define como **objetivo general**: incorporar las funcionalidades de los Módulos Análisis y Simulación del software alasBioSyS a la Plataforma de Servicios Bioinformáticos, del cual se derivan los siguientes **objetivos específicos**:

1. Incorporar el Módulo Simulación a la Plataforma de Servicios Bioinformáticos.
2. Incorporar el Módulo Análisis a la Plataforma de Servicios Bioinformáticos.
3. Validar el portlet y los servicios web implementados.

Para dar respuesta a estos objetivos se trazaron las siguientes **Tareas de la investigación**:

1. Realización de un estudio sobre el desarrollo de portlets.
2. Realización de una revisión bibliográfica sobre el desarrollo de servicios web.
3. Desarrollo del servicio web de la funcionalidad Simular del software alasBioSyS.
4. Desarrollo del servicio web de la funcionalidad Análisis de Bifurcaciones del software alasBioSyS.
5. Desarrollo del servicio web de la funcionalidad Análisis de Bifurcaciones en 1D/2D del software alasBioSyS.
6. Desarrollo del servicio web de la funcionalidad Análisis de Estabilidad del software alasBioSyS.
7. Desarrollo del servicio web de la funcionalidad Análisis por Reglas del software alasBioSyS.
8. Desarrollo del portlet de la aplicación.
9. Desarrollo de la interfaz para visualizar los resultados de los análisis realizados.
10. Validación del portlet y servicios web desarrollados mediante pruebas de caja negra y caja blanca.

El presente trabajo está estructurado en cuatro capítulos los cuales están resumidos a continuación:

Capítulo 1. Fundamentación teórica: Este capítulo está dedicado al estudio del arte del tema, así como al marco teórico. Se presentan la metodología de desarrollo y herramientas a utilizar. Se enuncian las tecnologías y el lenguaje de modelado.

Capítulo 2. Características del sistema: En este capítulo se profundizará sobre las funcionalidades que debe tener el sistema. Se presentan los requisitos funcionales y no funcionales. Además se muestran los casos de usos del sistema y su descripción, así como los actores que interactúan.

Capítulo 3. Análisis y Diseño de la aplicación: Se desarrollan los diagramas de clases del diseño, los patrones empleados, así como la realización de los diagramas de secuencia.

Capítulo 4. Implementación y Prueba: En este capítulo se exponen los diagramas de componentes, se describen las principales implementaciones realizadas en el software. Además se presentan las pruebas realizadas a la aplicación para validar la solución propuesta.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

Este capítulo está dedicado a la búsqueda y análisis de información sobre las tecnologías y los servicios web, así como el uso de portlets. También se presentan algunos de los portales bioinformáticos más utilizados actualmente. Se describen las herramientas y metodología de desarrollo utilizadas para la solución óptima de la problemática.

1.2 Tecnologías web

Las tecnologías web sirven para acceder a los recursos de conocimiento disponibles en Internet utilizando un navegador. Están muy extendidas por muchas razones: su flexibilidad en términos de escalabilidad, su sencillez de uso y que imitan la forma de relacionarse de las personas, al poner a disposición de todos el conocimiento de los demás, por encima de jerarquías, barreras formales u otras cuestiones, son fáciles de personalizar y construir[5].

Bajo el concepto de tecnología web se agrupan una serie de nuevas tecnologías y estándares que hacen posible Internet:

- **HTML (Hypertext Markup Language)** es el lenguaje básico para hacer archivos de texto con hipervínculos, publicarlos en un servidor web para ser usado por diversos usuarios, que pueden leerlos gracias a los navegadores web.
- **TCP/IP (Transport Control Protocol/Internet Protocol)** es el protocolo de comunicación más utilizado de Internet e Intranets, admite la comunicación punto a punto entre diferentes computadoras de una red.
- **HTTP (Hypertext Transfer Protocol)** es el protocolo web que gestiona las peticiones y servicios de documentos HTML.
- **FTP (File Transfer Protocol)** es un protocolo que sirve para transferir archivos a través de Internet. Por lo general, FTP se usa para poner archivos a disposición de otras personas para que puedan descargarlos, pero también se puede usar para cargar páginas web durante la creación de un sitio.

1.3 Portal Web

Un portal web es un sitio web que proporciona un único punto de interacción con aplicaciones, información, personas y procesos. Además son personalizados a las necesidades y responsabilidades

del usuario. Es un conjunto de páginas web que ofrecen información, herramientas y/o servicios a sus usuarios, de esta manera se le brinda al usuario, la facilidad de poder encontrar en este sitio todas sus necesidades sin salir del mismo[6].

Provee una interfaz común a una comunidad de usuarios determinada para asegurar un único punto de acceso a los servicios y recursos de su interés, entre los que se pueden encontrar: chats, foros, buscadores, compra electrónica, correo electrónico, aplicaciones, informaciones, entre otros. Pueden además ofrecer vistas personalizadas a usuarios específicos. Generalmente en los portales se agrupan la información de interés categorizando el contenido.

También es un punto de entrada a Internet donde se organizan sus contenidos, ayudando al usuario y concentrando servicios y productos; y que pueda encontrar allí todo cuanto utiliza a diario sin tener que salir de dicho sitio web[7].

1.4 Portales Bioinformáticos

En la actualidad se cuenta con numerosos portales dedicados al estudio bioinformático, donde se pueden realizar varias operaciones y servicios. Los más utilizados en la actualidad por los investigadores a nivel mundial se presentan a continuación. Estos portales son actualizados diariamente y cuentan con información desde hace varias décadas. Además existe una colaboración internacional entre las 3 principales bases de datos de nucleótidos, la EMBL-Bank en el EBI, el DDBJ en el CIB/NIG y el GenBank en el NCBI. Estas bases de datos intentan alojar todas las secuencias de nucleótidos que son de dominio público.

El Banco de Datos de ADN de Japón (DDBL) es el único banco de datos de secuencia de nucleótidos en Asia, que tiene la certificación oficial para recoger las secuencias de nucleótidos de los investigadores y para emitir el número de acceso de reconocimiento internacional a los que envían datos. Intercambia los datos recogidos con EMBL-Bank/EBI, el Instituto Europeo de Bioinformática y el GenBank / NCBI[8].

El Centro Nacional para la Información Biotecnológica (NCBI) cuenta con sistemas automatizados de almacenamiento y análisis de los conocimientos sobre la Biología Molecular, la Bioquímica y la Genética, facilitando el uso de dichas bases de datos y el software de la investigación entre la comunidad médica, coordinando los esfuerzos por reunir la información biotecnológica tanto a nivel nacional como internacional[9].

El Expasy es el Portal de Recursos Bioinformáticos del Swiss Institute of Bioinformatics (SIB), que proporciona acceso a bases de datos científicas y herramientas de software, en diferentes áreas de ciencias de la vida, incluida la Proteómica, la Genómica, Filogenia, Biología de sistemas, Genética de poblaciones, Transcriptómica, entre otras[10].

El Instituto Europeo de Bioinformática (EBI) está dedicado a la investigación académica, se encuentra en el Wellcome Trust Genoma Campus en Hinxton, cerca de Cambridge (Reino Unido), que forma parte de la European Molecular Biology Laboratory (EMBL). Entre los objetivos principales tiene:

- Proporcionar datos de libre acceso y servicios de Bioinformática para todas las facetas de la comunidad científica en formas que promuevan el progreso científico.
- Contribuir al avance de la Biología básica a través de la investigación impulsada por la Bioinformática.
- Proporcionar formación avanzada de la Bioinformática para científicos de todos los niveles, desde estudiantes de doctorado a los investigadores independientes.
- Ayudar a difundir las tecnologías de vanguardia para la industria[11].

1.5 Plataforma de Servicios Bioinformáticos

Esta plataforma está concebida y pensada en la Universidad de las Ciencias Informáticas. El objetivo de este sistema soportado sobre una Grid es permitir a los biólogos acceder a características avanzadas, como son los flujos de trabajo y el uso de ontologías para la composición y ejecución de tareas, sin tener que manejar conceptos complejos, ni instalar programas sofisticados[12].

Esta plataforma trabaja de forma general cuando el usuario a través del navegador web puede realizar una petición al servidor mediante el protocolo HTTP. Dicha petición es gestionada y enviada al contenedor de portlets, el cual en función del servicio solicitado, muestra los portlets correspondientes. Cuando el servicio a ejecutarse dentro del portlet requiere gran capacidad de cómputo se utiliza la plataforma de cálculos distribuidos T-Arenal para realizar los cálculos de manera eficiente.

Por último cada uno de los componentes se muestra al usuario cumpliendo las especificaciones estándares de portlets JSR 168 y 286; y la información referente a los perfiles de usuarios y configuración de los mismos es gestionada en una base de datos de PostgreSQL garantizando así la persistencia de los datos.

1.6 Servicios Web

Los servicios web son aplicaciones auto-contenidas, auto-descriptivas y modulares, que pueden ser publicadas, localizadas e invocadas a través de la web y que cuentan con un mecanismo estándar para establecer la comunicación con otros tipos de software a través de la red.

1.6.1 Características y operaciones de los servicios web

El esquema de funcionamiento de los servicios web, requiere de tres elementos fundamentales:

1. Un proveedor del servicio web, que es quien lo diseña, desarrolla e implementa y lo pone disponible para su uso, ya sea dentro de la misma organización o en público.
2. Un consumidor del servicio, que es quien accede al componente para utilizar los servicios que éste presta.
3. Un agente de servicio, que sirve como enlace entre proveedor y consumidor para efectos de publicación, búsqueda y localización del servicio.

Un servicio web simple está caracterizado por cuatro estándares: XML, SOAP, UDDI y WSDL, los cuales al trabajar juntos proporcionan una funcionalidad básica de tipo “solicitud/respuesta”[13].

1.7 Portlets

La especificación Java Portlet (JSR 168) define los portlets como componentes web basados en Java, administrados por un contenedor de portlets, que procesa solicitudes y genera contenido dinámico.

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

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

1.7.1 Funcionalidades adicionales que proporcionan los portlets

- **Almacenamiento persistente para las preferencias:** Los portlets proporcionan un objeto PortletPreferences para almacenar las preferencias de usuario. Estas preferencias son

almacenadas en una base de datos persistente, así se encontrarán disponibles cada vez que el contenedor de portlets se reinicie.

- **Procesamiento de solicitudes:** Los portlets disponen de una manipulación de peticiones más refinada. Pueden obtener su solicitud cuando el usuario hace alguna acción sobre éste.
- **Modos de los Portlets:** Los portlets usan el concepto de "mode" para indicar qué está haciendo el usuario. Los portlets normalmente se encuentran en un modo Vista (VIEW). Hay otras actividades, como especificar el tiempo de actualización o la configuración de datos como el nombre de usuario y la contraseña, por lo que se encuentran bajo el modo Editar (EDIT). La funcionalidad de ayuda se enmarca sobre el modo de Ayuda (HELP).
- **Estado de la ventana:** El estado de una ventana determina la cantidad de espacio que podría asignársele al contenido generado por un portlet sobre el portal. Si se pulsa en el botón maximizar el portlet utiliza todo el espacio disponible en la pantalla, de igual forma si éste pasa a estado minimizado únicamente se mostrará la barra de título asociada al portlet[14].

1.8 alasBioSyS

La herramienta alasBioSyS tiene como objetivo fundamental realizar exploraciones intensivas sobre modelos de sistemas biológicos para luego aplicar técnicas de meta análisis sobre los resultados arrojados[15].

Este software cuenta con diversas herramientas distribuidas en diferentes módulos:

- **Base de Datos:** La base de datos con que cuenta alasBioSyS permite almacenar los modelos matemáticos en formatos MathML y MATLAB, las condiciones iniciales y parámetros de dichos modelos, así como toda la información relacionada con los resultados que se obtienen al realizar exploraciones intensivas sobre estos.
- **Módulo de Simulación:** Las exploraciones se realizan utilizando funcionalidades brindadas por este módulo, permitiendo realizar simulaciones sencillas hasta grupos de simulaciones que se pueden distribuir utilizando una plataforma de cálculo distribuido.
- **Módulo de Análisis:** Este módulo permite realizar meta análisis sobre los resultados de las simulaciones. Cuenta con los siguientes métodos de análisis:
  - ✓ Dinámica de poblaciones.
  - ✓ Algoritmos de agrupamiento.
  - ✓ Clasificación.
  - ✓ Análisis por reglas.

- **Editor de ecuaciones:** Permite que los usuarios que interactúan con la herramienta puedan editar los sistemas de ecuaciones que describen los modelos que desean estudiar a través de una interfaz amigable. Garantiza además que la estructura sintáctica de las ecuaciones sea correcta y brinda mecanismos que permitan realizar un análisis de homogeneidad dimensional de las mismas.

1.9 Metodologías de desarrollo

Desde el surgimiento del desarrollo informático los desarrolladores han sido desordenados en la realización de productos tecnológicos, algunos por desconocimiento y otros por no emplear las metodologías relacionadas al desarrollo de aplicaciones.

En todo proceso de software se definen las metodologías de desarrollo que serán más ventajosas para la aplicación, para garantizar la calidad en el ciclo de vida del desarrollo del software. Entre las metodologías más utilizadas en el mundo se encuentran RUP, OpenUP y XP.

RUP

El Rational Unified Process (RUP) es un proceso de ingeniería de software. Proporciona un enfoque disciplinado para la asignación de tareas y responsabilidades dentro de una organización. Su objetivo es garantizar la producción de software de alta calidad que satisfaga las necesidades de los usuarios finales, dentro de un horario predecible y el presupuesto asignado[16].

Características Principales de RUP:

- Unifica los mejores elementos de metodologías anteriores.
- Preparado para desarrollar grandes y complejos proyectos.
- Orientado a Objetos.
- Utiliza el UML como lenguaje de representación visual.

Principales ventajas de RUP:

- Reduce el riesgo de no sacar el producto en el calendario previsto.
- Acelera el ritmo de desarrollo.
- Se adapta mejor a las necesidades del cliente.

Extreme Programming (XP)

Hace hincapié en la satisfacción del cliente. Posibilita entregar el software en una fecha temprana. La metodología XP se centra en el trabajo en equipo.

Los administradores, clientes y desarrolladores son socios iguales en un equipo de colaboración. Implementa un entorno simple, pero eficaz; permite a los equipos ser altamente productivos. El equipo se auto-organiza en torno al problema para resolverlo lo más eficientemente posible. Extreme Programming mejora un proyecto de software en cinco aspectos esenciales: la comunicación, la sencillez, la retroalimentación, el respeto y el coraje. Los programadores extremos constantemente se comunican con sus clientes y colegas programadores[17].

OpenUP

OpenUP es un proceso de desarrollo iterativo del software que es mínimo, completo, y extensible. El proceso es mínimo en que solamente el contenido fundamental es incluido; es completo en que puede ser manifestado como todo el proceso para construir un sistema; extensible en que puede ser utilizado como fundamento sobre el cual el contenido del proceso se pueda agregar o adaptar según lo necesitado[18].

Beneficios en el uso del OpenUP

- Es apropiado para proyectos pequeños y de bajos recursos. Esto permite disminuir las probabilidades de fracaso en los proyectos pequeños e incrementar las probabilidades de éxito.
- Permite detectar errores en etapas tempranas de un ciclo iterativo.
- Evita la elaboración de documentación, diagramas e iteraciones innecesarios requeridos en la metodología RUP.

Por todo lo anteriormente expuesto y estar definido como metodología en el proyecto del Portal de Servicios Bioinformáticos así como en el módulo alasBioSyS se escoge a OpenUP como metodología de desarrollo de software.

1.10 Lenguaje de modelado

El Lenguaje Unificado de Modelado (UML por sus siglas en inglés), es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software.

El proceso unificado de desarrollo de software representa un número de modelos de desarrollo basados en componentes que han sido propuestos en la industria. Utilizando el UML, el proceso unificado define los componentes que se utilizarán para construir el sistema y las interfaces que conectaran los componentes[19].

Se utilizará la versión 2.0 para modelar la propuesta del software.

1.11 Herramientas CASE

Se puede definir a las Herramientas CASE (Computer-Aided Software Engineering) como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida del desarrollo de un software[20].

Platinum ERwin

Platinum ERwin es una herramienta para el diseño de bases de datos y permite generar y dar mantenimiento de aplicaciones. ERwin permite visualizar la estructura, los elementos importantes, y optimizar el diseño de la base de datos. Genera automáticamente las tablas y miles de líneas de "stored procedure" y "triggers" para los principales tipos de base de datos. Esta herramienta es soportada por diversos manejadores de bases de datos, como Oracle, SQL Server, Sysbase, entre otros[21].

Rational Rose

El Rational Rose es una herramienta que permite crear los diagramas que se van generando durante el proceso de Ingeniería en el Desarrollo del Software.

Entre sus principales características se encuentran:

- Soporte para análisis de patrones ANSI C++, Rose J y Visual C++.
- Soporte de ingeniería Forward y/o reversa para algunos de los conceptos más comunes de Java.
- La generación de código Ada, ANSI C ++, C++, CORBA, Java y Visual Basic, con capacidad de sincronización modelo- código configurables.
- Soporte Enterprise Java Beans 2.0.
- Capacidad de análisis de la calidad del código[22].

Visual Paradigm

Visual Paradigm tiene como objetivo fundamental reducir el tiempo de desarrollo de analistas, arquitectos, diseñadores y desarrolladores en la concepción de un software.
Soporta el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas. Constituye una herramienta de software libre de probada utilidad para el analista.

Se caracteriza por:

- Disponible en múltiples plataformas (Windows, Linux).
- Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Posee capacidades de ingeniería directa e inversa.
- Presenta dos tipos de licencia: gratuita y comercial.
- Soporta aplicaciones web.
- Importa y exporta de ficheros XMl.
- Contiene un editor de figuras.

Se decide utilizar esta herramienta en su versión 8.0 por ser multiplataforma y por la universidad contar con una licencia para el uso de la misma.

1.12 Herramientas y tecnologías de desarrollo

1.12.1 Lenguajes de Programación del lado del servidor

El diccionario de la RAE define los lenguajes de programación como “un conjunto de caracteres, símbolos, representaciones y reglas que permiten introducir y tratar la información en un ordenador”.

En la actualidad existen diferentes lenguajes de programación para el desarrollo de aplicaciones web. Los mismos se pueden clasificar en lenguajes del lado del cliente o lenguajes del lado del servidor. Entre los más utilizados se encuentran PHP, C-Sharp y Java.

PHP

PHP, es un lenguaje de código abierto, originalmente diseñado para ser usado en el desarrollo de aplicaciones web. PHP no necesita ser compilado, y el código está embebido en las páginas HTML por

lo que el proceso de cargar las páginas es bastante corto. Se puede obtener sin necesidad de pagar ninguna licencia, es multiplataforma, posee marcos de trabajo bien conocidos como Symphony, ZendFramework y PHPNuke y también cuenta con Sistemas de Manejo de Contenidos (CMS por sus siglas en inglés) muy populares como Drupal, Joomla y WordPress[23].

C-Sharp o C#

La compañía Microsoft proporciona una plataforma de desarrollo web denominada ASP.NET. Abarca un gran número de lenguajes de programación entre los que se encuentran C, C# y C++. Incluye varios controles prefabricados, como cajas de texto, botones, imágenes, y datos cuadriculados, que pueden ser ensamblados, configurados y manipulados con código para crear páginas HTML procesadas por los navegadores web[24].

Java

Una de las principales características de Java es la de ser un lenguaje compilado e interpretado. Todo programa en Java ha de compilarse y el código que se genera en "bytecodes" es interpretado por una máquina virtual. De este modo se consigue la independencia de la máquina, el código compilado se ejecuta en máquinas virtuales que si son dependientes de la plataforma[25].

Este lenguaje posee marcadas características entre las que se puede mencionar:

- **Orientado a objeto:** Java trabaja con sus datos como objetos y con interfaces a esos objetos. Soporta las tres características propias del paradigma de la orientación a objetos: encapsulamiento, herencia y polimorfismo.
- **Robusto:** Realiza verificaciones en busca de problemas tanto en tiempo de compilación como en tiempo de ejecución.
- **Seguro:** Las aplicaciones de Java resultan extremadamente seguras, ya que no acceden a zonas delicadas de memoria o de sistema.

Considerando lo anteriormente planteado se decide utilizar el lenguaje de programación Java para la solución del problema planteado.

1.12.2 Lenguajes de programación del lado del cliente

JavaScript

JavaScript es un lenguaje de script multiplataforma, pequeño, ligero y orientado a objetos. No es útil como un lenguaje independiente, más bien está diseñado para una fácil incrustación en otros productos y aplicaciones, tales como los navegadores web. Dentro de un entorno anfitrión, Javascript puede ser conectado a los objetos de su entorno para proveer un control programable sobre éstos[26].

Se utilizará JavaScript para la validación del lado del cliente de las páginas web.

CSS

El W3C define CSS como "un mecanismo simple para añadir estilo (fuentes, colores, espacios, imágenes) a documentos Web". Es un lenguaje de diseño estándar para la web, reduce la sobrecarga del servidor y acelera la carga de las páginas web.

1.12.3 Herramientas para el desarrollo

Eclipse Helios

Eclipse Foundation es una comunidad de código abierto, donde sus programadores se enfocan en el perfeccionamiento de una plataforma de desarrollo abierta formada por marcos extensibles, herramientas y tiempos de ejecución para la construcción, implementación y administración de software a través del ciclo de vida. Eclipse Helios es la liberación anual de proyectos de Eclipse en el 2010[27].

Ventajas de Eclipse:

- Emplea módulos (en inglés plug-in) para proporcionar toda su funcionalidad al frente de la Plataforma de Cliente Rico, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no.
- Al permitirle a Eclipse extenderse usando otros lenguajes de programación como son C/C++ y Python, admite trabajar con lenguajes para procesado de texto como LaTeX, aplicaciones en red como Telnet y Sistemas de Gestión de Bases de Datos.
- La arquitectura plug-in permite escribir cualquier extensión deseada en el ambiente, como sería Gestión de la Configuración.

Se optó por usar Eclipse Helios, ya que en la práctica resultó ser más económico en la utilización de los recursos de hardware y por lo tanto se muestra con mayor rendimiento y rapidez que la versión 7.1 de NetBeans.

1.12.4 Herramientas y marcos de trabajo para el desarrollo de servicios web y portlets

1.12.4.1 Contenedores de portlets

En la plataforma J2EE existen varios contenedores de portlets que son capaces de manejar el ciclo de vida de un portlet, entre los más utilizados están Apache Jetspeed, JBoss Portal y Liferay.

Apache Jetspeed

Los datos presentados a través de Jetspeed son independientes del tipo de contenido. Esto significa que contenidos como XML, RSS o SMTP se pueden integrar a Jetspeed. La presentación actual de los datos se maneja a través de XSL y se entregan al usuario, por ejemplo, a través de la combinación de Java Server Pages (JSP) y HTML[28].

JBoss Portal

JBoss Portal proporciona una plataforma de código abierto para acoger y servir a la interfaz de un portal web, así como la publicación y gestión de su contenido. JBoss es un servidor de aplicaciones J2EE implementado en Java, esto posibilita que pueda ser utilizado en cualquier sistema operativo para el que esté disponible Java[29].

Liferay

Liferay Portal es un portal de gestión de contenidos de código abierto implementado sobre Java y su contenido está basado en Portlets. Liferay Portal incluye muchas aplicaciones portlets tales como mensajería instantánea, foros y biblioteca de documentos. Soporta múltiples base de datos como: PostgreSQL, MySQL, Oracle, SQL Server, Sybase e InterBase, se puede desplegar con muchos servidores como Jetty, JBoss, Sun GlassFish, Oracle AS y Apache Tomcat, es multiplataforma ya que puede ejecutarse en cualquier sistema operativo como Windows, Linux y MacOS[30].

Se utilizará la versión de Liferay 6.0.5 como contenedor de portlets.

1.12.4.2 Herramientas para el trabajo con servicios web

Apache Tomcat

Tomcat es un producto muy robusto, altamente eficiente y uno de los más potentes contenedores de Servlets existentes. Su único punto débil reside en la complejidad de su configuración, dado el gran número de opciones existentes[31].

Características

- Apache Tomcat es un servidor HTTP y un contenedor de servlets.
- Es la implementación de referencia de las especificaciones de servlets (2.4) y de JSP (2.0).
- Es software libre (licencia Apache 2.0) gestionado por la fundación Apache.
- Puede funcionar como servidor HTTP o conectado a otro servidor HTTP como Apache HTTP Server o IIS[32].

Se utilizará la versión 6.0.26 para el despliegue de los servicios web.

Apache Axis 2

Apache Axis 2 es un motor de servicios Web implementado en el lenguaje de programación C. Se basa en la arquitectura Axis 2 extensible y flexible. Apache Axis 2 se puede utilizar para proporcionar y consumir servicios web. Se ha implementado la portabilidad y la capacidad de incrustar en la mente, por lo tanto, podría ser utilizado como un habilitador de servicios web en otro software[33].

1.12.4.3 Marcos de trabajo

Actualmente la plataforma J2EE cuenta con algunos marcos de trabajo para el desarrollo de aplicaciones web, entre las más populares se encontraron a Struts, Java Server Faces y Spring Framework.

Struts

Struts es una herramienta de soporte para el desarrollo de aplicaciones Web bajo el patrón MVC y la plataforma Java EE (Java Enterprise Edition). Struts se desarrollaba como parte del proyecto Jakarta de la Apache Software Foundation, pero actualmente es un proyecto independiente conocido como Apache Struts.

Struts permite reducir el tiempo de desarrollo. Su carácter de "software libre" y su compatibilidad con todas las plataformas en las que Java Enterprise esté disponible, esto lo convierten en una herramienta altamente disponible[34].

Java Server Faces (JSF)

Java Server Faces es un marco de trabajo de desarrollo web basado en Java. JSF está incluido en la plataforma Java EE, por lo que puede crear aplicaciones que utilizan JSF sin añadir bibliotecas adicionales en el proyecto[35].

La tecnología Java Server Faces establece el estándar para la construcción interfaces de usuario del lado del servidor. Con las aportaciones del grupo de expertos, las API JavaServer Faces se están diseñando para que puedan ser aprovechadas por las herramientas que harán aún más fácil el desarrollo de aplicaciones web[36].

Spring

Spring es un marco de trabajo de código abierto. Permite el desarrollo de aplicaciones J2EE, suscitando buenas prácticas de diseño y programación entre los usuarios. Una de las ventajas que posee está la modularidad, permitiendo hacer uso de algunos de los módulos sin implicarse con el uso del resto.

Módulos principales de Spring

- Spring Core (Inversión del control (IoC) / Inyección de dependencias (DI)).
- Spring AOP (Programación orientada a aspectos).
- Spring JDBC (Acceso a datos).
- Spring MVC (desarrollo Web según el patrón MVC).
- Spring Remoting (distribución).
- Spring Transaction Abstraction (Gestión de transacciones).
- Otros: Spring TestContext (pruebas), Spring JMX (gestión), Spring Security (seguridad), Spring Web Flow (control de flujos en aplicaciones Web)[37].

Ventajas de Spring MVC

- Ciclo de vida para anular enlaces y validaciones.
- Se integra con muchas opciones de visualización sin problemas: JSP, Tiles, Velocity, FreeMarker, Excel, PDF.
- El control de versiones hace que sea fácil comprobar[38].

1.13 Gestor de Base de Datos PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado y en sus últimas versiones no tiene nada que envidiarle a otras bases de datos comerciales[39].

Posee interfaces nativas para lenguajes como JDBC, C, C++, PHP, PERL, TCL, ECPG, PYTHON y RUBY. Funciona en los principales sistemas operativos: Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64) y Windows.
Debido a la liberación de la licencia, PostgreSQL se puede usar, modificar y distribuir de forma gratuita para cualquier fin, ya sea privado, comercial o académico. Será utilizada para el manejo de los datos la versión 8.4.

1.14 Conclusiones

En el presente capítulo se describieron los principales conceptos relacionados con el desarrollo de aplicaciones web, así como las características más representativas. Se presentaron los portales bioinformáticos más utilizados actualmente en el mundo, estos se actualizan diariamente y mantienen una sincronización de datos que los ubica como líderes mundiales.
Se usarán los servicios web, mediante el lenguaje de programación Java, el cual permitirá el uso de componentes web como los Servlets, JSP y Portlets. Será usada como herramienta contenedora de portlets el Liferay y el Apache Tomcat y Axis 2 para el despliegue de los servicios web. Se definió al software de desarrollo Eclipse Helios para la implementación del problema propuesto. Como marco de trabajo se estableció el uso de Spring MVC.
El proceso de desarrollo del software está guiado por la metodología OpenUP, la misma permite la representación mediante diagramas del proceso de concepción de la aplicación informática; utiliza como lenguaje de modelado a UML y la herramienta de CASE: Visual Paradigm.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

2.1 Introducción

El presente capítulo muestra el modelo de dominio con la relación entre las clases, así como la especificación de los requisitos funcionales y no funcionales de la aplicación. Presenta el diagrama de casos de usos del sistema para identificar la relación entre los actores y casos de usos del sistema.

2.2 Modelo de dominio

El modelo de dominio (Figura 1) es una representación gráfica del problema mediante el enlace de clases, que no identifican componentes del software, sino objetos del mundo real.

El análisis orientado a objetos tiene por finalidad estipular una especificación del dominio del problema y los requisitos desde la perspectiva de la clasificación por objetos y desde el punto de vista de entender los términos empleados en el dominio. Para descomponer el dominio del problema hay que identificar los conceptos, los atributos y las asociaciones del dominio que se juzgan importantes[40].

Como consecuencia de la pobre identificación de los procesos del negocio y para mejor entendimiento del funcionamiento final de la herramienta, se utilizará modelo de dominio.

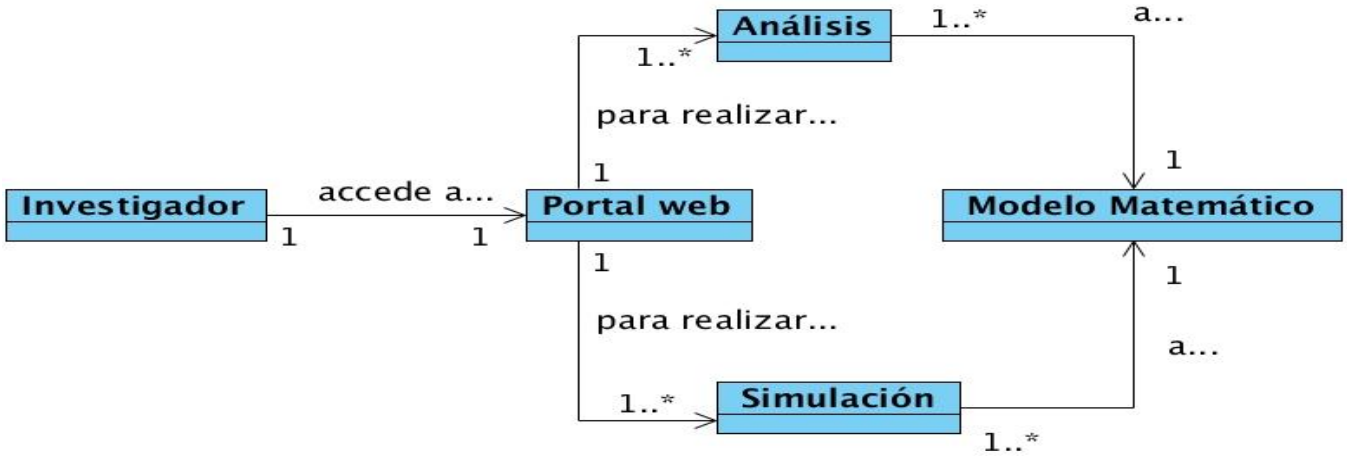


Fig. 1: Diagrama de Clases del Modelo de Dominio de la aplicación. Representa los conceptos del mundo real y nunca los relacionados con objetos de software.

Tabla 1: Descripción de las clases del Modelo de Dominio.

Investigador	Persona habilitada para trabajar con el software.

Modelo Matemático	Conjunto de ecuaciones diferenciales que describen un sistema biológico.
Simulación	Clase que representa la simulación de un modelo matemático.
Análisis	Clase que representa el análisis de un modelo matemático.
Portal web	Clase que representa un sitio web.

2.3 Modelado del sistema

2.3.1 Requisitos funcionales

Los requisitos funcionales de una aplicación son las capacidades o condiciones que el sistema debe cumplir para satisfacer las necesidades primarias del cliente.
A continuación se definen los requisitos funcionales seleccionados para dar cumplimiento al desarrollo de la aplicación. Los mismos están orientados al trabajo en la Plataforma de Servicios Bioinformáticos, que lo diferencian en relación a la aplicación de escritorio del software alasBioSyS.

- RF1: Realizar simulación.
- RF2: Realizar análisis.
  - ✓ RF2.1: Mostrar Dinámicas Aleatorias.
  - ✓ RF2.2: Realizar Análisis por reglas.
    - RF2.2.1: Definir valores necesarios para realizar el análisis.
  - ✓ RF2.3: Realizar Análisis de Bifurcaciones.
  - ✓ RF2.4: Realizar Análisis de Bifurcaciones en 1D/2D.
  - ✓ RF2.5: Realizar Análisis de Estabilidad.
  - ✓ RF2.6: Visualizar el estado de las ejecuciones que se realizan sobre T-Arenal.
  - ✓ RF2.7: Visualizar gráficas de Bifurcaciones 1D/2D hechas en T-Arenal.
  - ✓ RF2.8: Eliminar las soluciones de las ejecuciones realizadas en T-Arenal.
  - ✓ RF2.9: Descargar las soluciones de las ejecuciones realizadas en T-Arenal.

2.3.2 Requisitos no funcionales

Los requisitos no funcionales son las propiedades o cualidades que el producto debe presentar. Estos requisitos definirán las características del producto final y muchas veces están implicados en el éxito final que se pueda alcanzar.

- ✓ *De software.*
  - Tener instalado en el Servidor de Aplicaciones:

- Gestor de Base de Datos: PostgreSQL.
- Herramienta matemática Matlab.
- Herramienta matemática Octave.
- Plataforma de Cálculo Distribuido T-Arenal.

➢ Al menos un navegador web instalado en la PC cliente.

✓ *De usabilidad.*

➢ Debe dar la posibilidad del uso de la herramienta por cualquier persona, capacitada en conocimientos de bioinformática.

✓ *Requerimientos legales.*

➢ Los requisitos legales y el derecho de autor serán registrados por la Universidad de las Ciencias Informáticas.

2.3.3 Definición de los Casos de Usos del Sistema

Un diagrama de casos de uso explica gráficamente un conjunto de casos de uso de un sistema, los actores y la relación entre éstos y los casos de uso[41]. Estos diagramas sirven para facilitar la comunicación con los futuros usuarios del sistema, y con el cliente, y resultan especialmente útiles para determinar las características necesarias que tendrá el sistema.
El diagrama de caso de uso para la solución propuesta se muestra en la Figura 2.

2.3.3.1 Actor del sistema

Los actores representan terceros fuera del sistema que interactúan con este. El actor es una entidad externa del sistema que de alguna manera participa en la historia del caso de uso. Por lo regular estimula el sistema con eventos de entrada o recibe algo de él. Conviene escribir su nombre con mayúscula en la narrativa del caso para facilitar la identificación[42].
Acorde a la situación en la que se desarrollará la herramienta se encontró el siguiente actor:

Tabla 2: Definición del actor del sistema.

Actor	Descripción
Investigador	Cualquier persona capacitada para interactuar con el software a través del Portal de Servicios Bioinformáticos.

2.3.3.2 Casos de Usos del Sistema

Siguiendo la estrategia empleada en los requisitos funcionales, los casos de usos que se presentan serán realizados en la Plataforma de Servicios Bioinformáticos, diferenciándose de la aplicación de escritorio del software alasBioSyS.

1. **CU** Simular.
2. **CU** Mostrar Dinámicas Aleatorias.
3. **CU** Realizar Análisis por Reglas.
4. **CU** Realizar Análisis de Bifurcaciones.
5. **CU** Realizar Análisis de Bifurcaciones en 1D/2D.
6. **CU** Visualizar gráficas de Bifurcaciones 1D/2D hechas en T-Arenal.
7. **CU** Realizar análisis de estabilidad.
8. **CU** Administrar las soluciones de T-Arenal.
9. **CU** Administrar las ejecuciones de T-Arenal.

2.3.3.3 Diagrama de Casos de Usos del Sistema

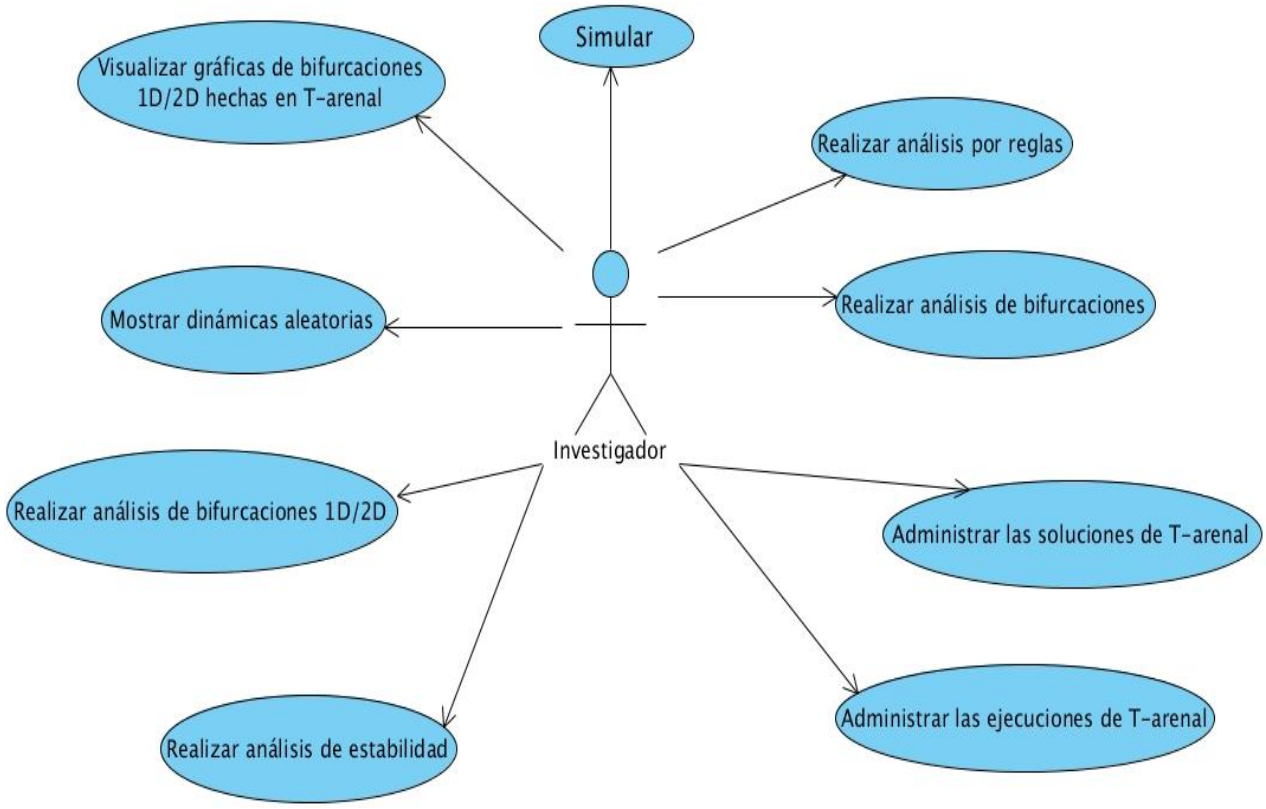


Fig. 2: Diagrama de Casos de Usos del Sistema. Los CU pueden contener en ocasiones varios requisitos funcionales. Los CU serán iniciados en todos los casos por el investigador.

2.3.4 Descripción de los Casos de Usos del Sistema

Tabla 3: Descripción del Caso de Uso Simular.

Caso de Uso	Simular.
Actores	Investigador (inicia)
Resumen	El investigador podrá someter un modelo a una simulación, mediante diferentes métodos de simulación, pudiendo obtener una o muchas simulaciones.
Precondiciones	Debe existir al menos un modelo matemático guardado en la base de datos.
Referencias	RF1
Prioridad	Crítico.
Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1. El investigador escoge la opción “Simulate” en la PSB.	
2. El investigador elige el modelo matemático que va a utilizar.	3. El sistema muestra la interfaz de simulación, donde el investigador puede introducir los siguientes valores: ✓ Valores de parámetros y poblaciones iniciales. ✓ Initial Time. ✓ Final Time. ✓ Relative Tolerance. ✓ Absolute Tolerance. ✓ Numerical Method. En caso de haber anteriormente seleccionado como herramienta matemática “ODEtoJava” se desplegarán los métodos numéricos correspondientes a la misma: ✓ DormandPrince. ✓ ErkSD. ✓ Erk. ✓ ErkTriple. ✓ Imex ✓ ImexSD

	En el caso de que se haya seleccionado como herramienta matemática “Octave” aparecerán como métodos matemáticos los siguientes: ✓ lsode-stiff. ✓ lsode-non-stiff. ✓ Ode23. ✓ Ode45. ✓ Ode78. ✓ Ode54. En el caso de que se haya seleccionado como herramienta matemática “Matlab” aparecerán como métodos matemáticos los siguientes: ✓ Runge-Kutta (2, 3). ✓ Runge-Kutta (4, 5). ✓ ode113 (adams). ✓ ode15s (stiff/NDF). ✓ ode23s (stiff/Mod. Rosenblock). ✓ ode23t (Mod. stiff/Trapezoidal). ✓ ode23tb (stiff/TR-BDF2).
4. El investigador introduce los valores necesarios correctamente para la simulación.	
5. El investigador escoge iniciar simulación.	6. El sistema comprueba que los datos estén correctos.
	7. El sistema salva los valores definidos.
	8. Comienza la simulación del modelo.
	9. El sistema muestra los resultados de la simulación.
	10. El sistema almacena los datos de la simulación y finaliza el CU.
Flujos alternos	
	6.1 Si los datos no son correctos el sistema lanza un error y finaliza el CU.

Pos condiciones	Se obtendrá un resultado de la simulación. El resultado de la simulación se almacenará en la base de datos.

El resto de las descripciones de los casos de uso, aparecen reflejadas en el Documento de apoyo de la tesis.

2.4 Conclusiones

En este capítulo fueron elaborados los artefactos correspondientes al proceso de desarrollo como el modelo de dominio, el cual permitió un mejor entendimiento del problema a resolver, mediante una representación gráfica de los factores que interactúan con la aplicación. Luego del estudio del problema se definieron 12 requisitos funcionales para refinarlos y convertirlos en casos de usos del sistema. También se presentaron los requisitos no funcionales, los que especifican las herramientas esenciales para el correcto funcionamiento del sistema, entre los que destacan MatLab y T-Arenal. Además se realizó la descripción del CU Simular por la importancia que reviste. Esta descripción posibilita la generación de artefactos en la metodología de desarrollo como los diagramas de interacción.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DE LA APLICACIÓN

3.1 Introducción

Este capítulo está dedicado a la descripción del estilo arquitectónico usado en el desarrollo de la herramienta. La descripción de los patrones de diseño utilizados así como los diagramas de clases del diseño. Se presentará el diagrama de secuencia y el diagrama de despliegue.

3.2 Arquitectura de Software

Sucede con mucha frecuencia que no se le presta especial atención a la asignación de responsabilidades y el uso incorrecto de los componentes de los diagramas de interacción. Una decisión errónea en estos aspectos puede desencadenar la realización de sistemas poco perdurables y sujetos a muchos cambios, perdiendo así tiempo y en muchos casos dinero con el cliente.
La arquitectura de un software define como va a estar estructurada la herramienta. Especifica el sistema de un software en términos de componentes computacionales y las interacciones entre los mismos.

3.3 Estilos y patrones arquitectónicos

La decisión por un estilo o patrón arquitectónico define el resultado final de una aplicación. Es necesario que el escogido sea aplicado correctamente para el desarrollo del software.
En la actualidad no existe un convenio para diferenciar patrones y estilos arquitectónicos[43].
Un estilo arquitectónico expresa componentes y las relaciones entre estos, con las restricciones de su aplicación y la composición asociada, así como también las reglas para su construcción.

Un patrón de arquitectura de software describe un problema particular y recurrente del diseño, que surge en un contexto específico, y presenta un esquema genérico y probado de su solución.
Definen la estructura de un sistema software, los cuales a su vez se componen de subsistemas con sus responsabilidades.
Se define como una regla que consta de tres partes, la cual expresa una relación entre un contexto, un problema y una solución.

- **Contexto:** Es una situación de diseño en la que aparece un problema de diseño
- **Problema:** Es un conjunto de fuerzas que aparecen repetidamente en el contexto
- **Solución:** Es una configuración que equilibra estas fuerzas. Ésta abarca:
  - Estructura con componentes y relaciones

- Comportamiento a tiempo de ejecución: aspectos dinámicos de la solución, como la colaboración entre componentes, la comunicación entre ellos[44].

3.3.1 Patrón Arquitectónico Modelo Vista Controlador (MVC).

Es un patrón de arquitectura de software que aparta los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes diferentes. El patrón MVC (Figura 3) se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página, el modelo es el Sistema de Gestión de Base de Datos y la Lógica de negocio y el controlador es el responsable de recibir los eventos de entrada desde la vista. La finalidad del modelo es mejorar la reusabilidad por medio del desacople entre la vista y el modelo[45].

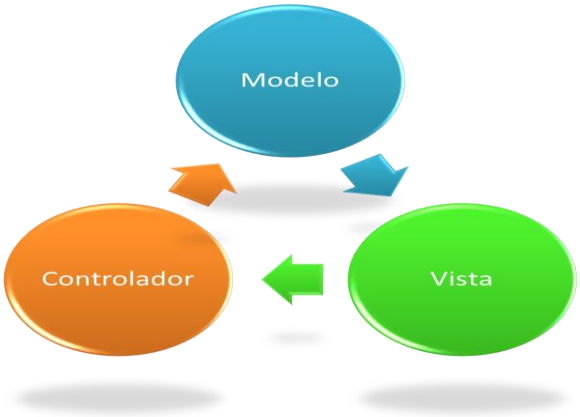


Fig. 3: Patrón Arquitectónico Modelo-Vista-Controlador. La utilización de este patrón evita en gran medida el acoplamiento de los componentes del sistema. Cada capa obedece a la solicitud de otra manteniendo una cohesión entre las mismas.

Modelo: Accede a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento. Define las reglas de negocio (la funcionalidad del sistema). Lleva un registro de las vistas y controladores del sistema.
Vista: Recibe datos del modelo y los muestra al usuario. Tienen un registro de su controlador asociado.
Controlador: Recibe los eventos de entrada (un clic, un cambio en un campo de texto, entre otros). Contiene reglas de gestión de eventos, del tipo “SI evento Z, entonces acción W”.

Entre las principales ventajas que presenta este estilo arquitectónico se encuentran:

- ✓ **Soporte para múltiples vistas:** la vista se separa del modelo y no existe dependencia directa entre vista y modelo, la interfaz de usuario puede exponer múltiples vistas de los mismos datos al mismo tiempo.

- ✓ **Mayor soporte a los cambios:** los requisitos de interfaz tienden a cambiar más rápidamente que las reglas de negocio. Puesto que el modelo no depende de las vistas, el aditamento de nuevos tipos de vista al sistema habitualmente no afectan al modelo. Por tanto, el ámbito del cambio se limita a la vista.

3.4 Patrones de diseño

Un Patrón de Diseño es una solución repetible a un problema recurrente en el diseño de software. Esta solución no es un diseño terminado que puede traducirse directamente a código, sino más bien una descripción sobre cómo resolver el problema[46].

Un patrón es un par problema/solución con nombre que se puede aplicar en nuevos contextos con consejos acerca de cómo aplicarlo en nuevas situaciones y discusiones sobre sus compromisos[47].

En resumen los patrones de diseños se pueden definir de la siguiente manera:

- Describen un problema recurrente y una solución.
- Cada patrón nombra, explica, evalúa un diseño recurrente en sistemas orientados a objetos.

En la realización del diseño para la aplicación informática a implementar, se utilizaron los patrones GRASP, es el acrónimo de General Responsability Assignment Software Patterns. Se considera que más que patrones son una serie de "Buenas Prácticas" de aplicación recomendable en el diseño de software.

3.4.1 Patrón Bajo Acoplamiento

Este patrón asigna la responsabilidad de controlar el flujo de eventos del sistema a clases específicas. Esto facilita la centralización de actividades. El controlador no las realiza, las delega en otras clases con las que mantiene un modelo de alta cohesión.

En el diseño del software, se pretende el acoplamiento más bajo posible. Una conectividad sencilla entre los módulos da como resultado un software más fácil de entender y menos propenso a tener un “efecto ola”, causado cuando ocurren errores en un lugar y se propagan por el sistema[48].
Este patrón se refleja en la Figura 4.

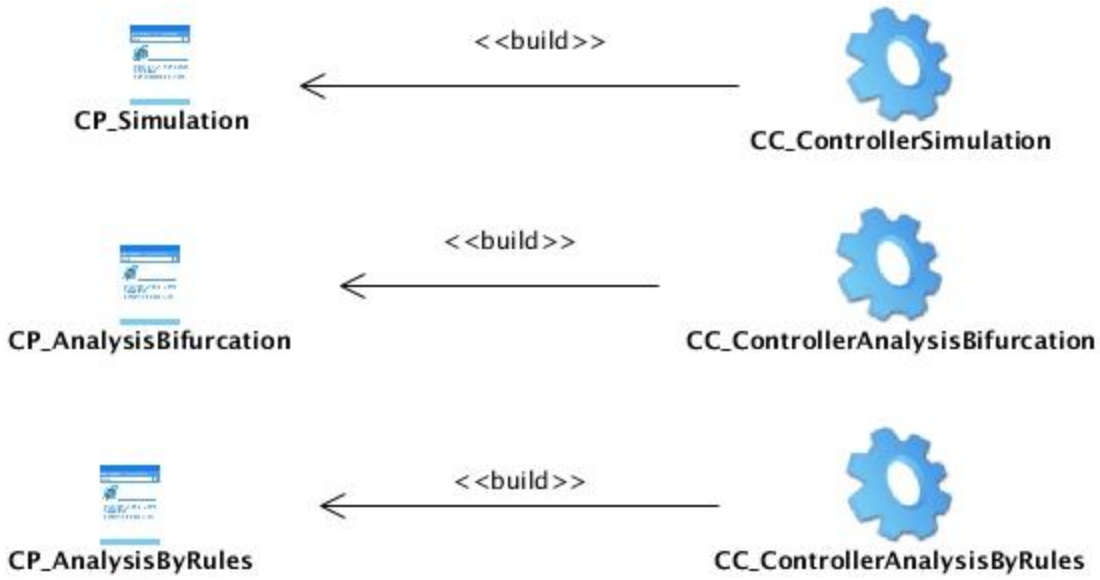


Fig. 4: Representación del Patrón Bajo Acoplamiento y Alta Cohesión en el Diseño del Sistema. Explica cómo dar soporte a una dependencia escasa y cómo asignar una responsabilidad de modo que la cohesión siga siendo alta respectivamente.

3.4.2 Patrón Alta Cohesión

El patrón alta cohesión describe como fuertemente los contenidos internos de una rutina están relacionados entre sí[49].

Este patrón propone asignar la responsabilidad de manera que la complejidad se mantenga dentro de límites manejables asumiendo solamente las responsabilidades que deben manejar, evadiendo un trabajo excesivo. Este patrón se evidencia en la figura 4.

3.4.3 Patrón Creador

El Patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que conecte con el objeto producido en cualquier evento[50].

La nueva instancia del objeto deberá ser creada por la clase que: Tiene la información necesaria para realizar la creación del objeto, usa directamente las instancias creadas del objeto, o almacena o maneja varias instancias de la clase. Este patrón brinda soporte de bajo acoplamiento, lo cual supone menos dependencias entre clases y posibilidades.

Se ejemplifica en la Figura 5.

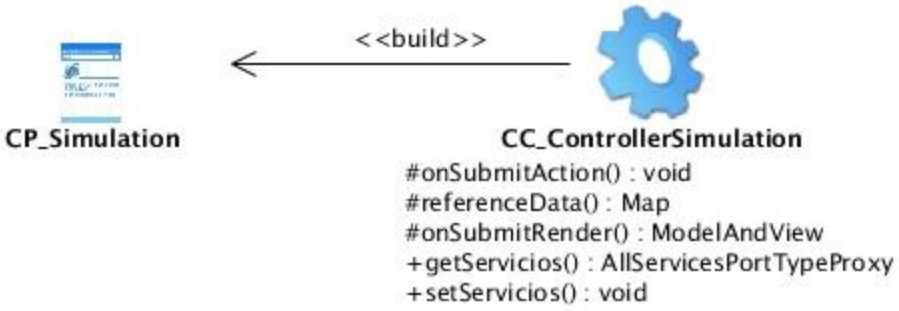


Fig. 5: Representación del Patrón Creador en el Diseño del Sistema. Explica que clase es la encargada de crear objetos, en determinados escenarios de ejecución.

3.4.4 Patrón Controlador

El Patrón Controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema a una clase que represente un sistema global. Define además el método de su operación[51].

Está representado en la figura 6.

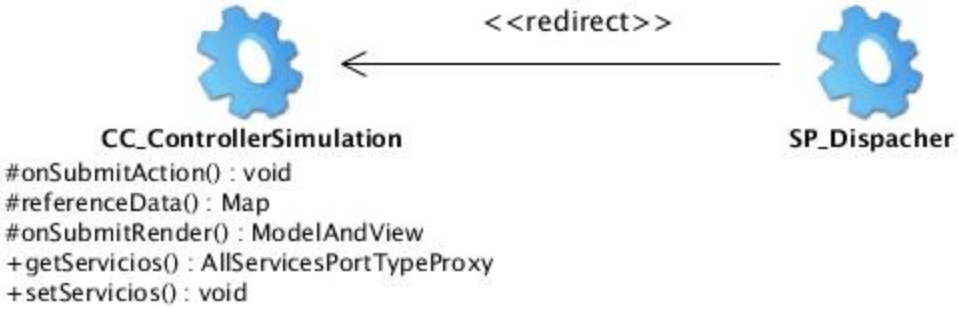


Fig. 6: Representación del Patrón Controlador en el Diseño del Sistema. Asigna la responsabilidad del manejo de los eventos de un sistema a una clase que represente un sistema global.

3.5 Modelo de diseño

El modelo de diseño detalla los modelos de análisis, tomando en cuenta todas las implicaciones y restricciones técnicas

El propósito es especificar una solución que trabaje y pueda ser fácilmente convertida en código fuente y construir una arquitectura simple y extensible. Las clases definidas en el análisis se detallan y se añaden nuevas clases para manejar áreas técnicas como bases de datos, interfaces de usuario, dispositivos, entre otros[52].

3.5.1 Diagrama de secuencia

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso.

Se indican los módulos o clases que forman parte del programa y las llamadas que se hacen en cada uno de ellos para realizar una tarea determinada. El detalle que se muestre en el diagrama de secuencia debe estar en consonancia con lo que se intenta mostrar o bien con la fase de desarrollo en la que esté el proyecto. En el diagrama de secuencia no se ponen situaciones erróneas, puesto que poner todos los detalles puede dar lugar a un diagrama que no se entiende o difícil de leer[53].

El diagrama de secuencia de la Figura 8 muestra la comunicación entre diferentes componentes del software. Se determinó una clase controladora principal “CC_Principal”, la cual gestionará todas las peticiones del usuario e intercambia peticiones con las demás clases.

A continuación se presenta el diagrama de secuencia del CU Simular. El resto de los diagramas de secuencia, aparecen reflejados en el Documento de apoyo de la tesis.

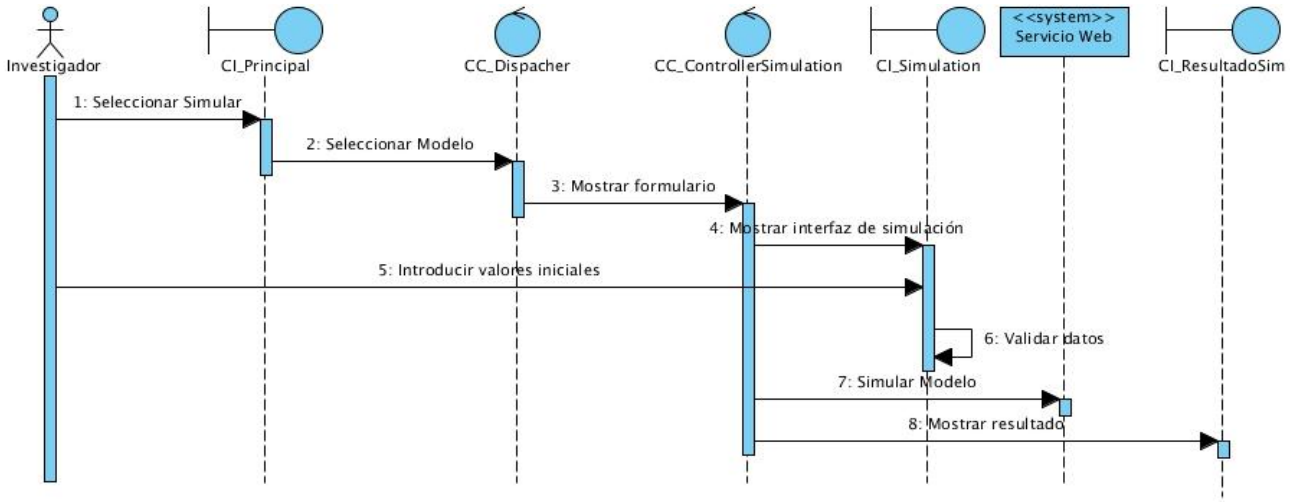


Fig. 7: Diagrama de Secuencia del CU Simular. Las clases interfaces (CI) realizan peticiones a las clases controladores (CC) y estas realizan las llamadas a los servicios web.

El resto de los diagramas de secuencia, aparecen reflejados en el Documento de apoyo de la tesis.

3.5.2 Diagrama de clases del diseño

Los diagramas de clases son utilizados durante el proceso de análisis y diseño de los sistemas, donde se crea el diseño conceptual de la información que se manejará en el sistema, y los componentes que se encargaran del funcionamiento y la relación entre uno y otro.

En el diagrama de clases del diseño para el CU Simular (Figura 7) se evidencia el empleo del patrón Modelo Vista Controlador divididos en tres paquetes, que simbolizan: la Presentación al usuario, la Lógica con la cual el sistema trabaja y los Servicios de Acceso a Datos.

El resto de los diagramas de clases del diseño, aparecen reflejados en el Documento de apoyo de la tesis.

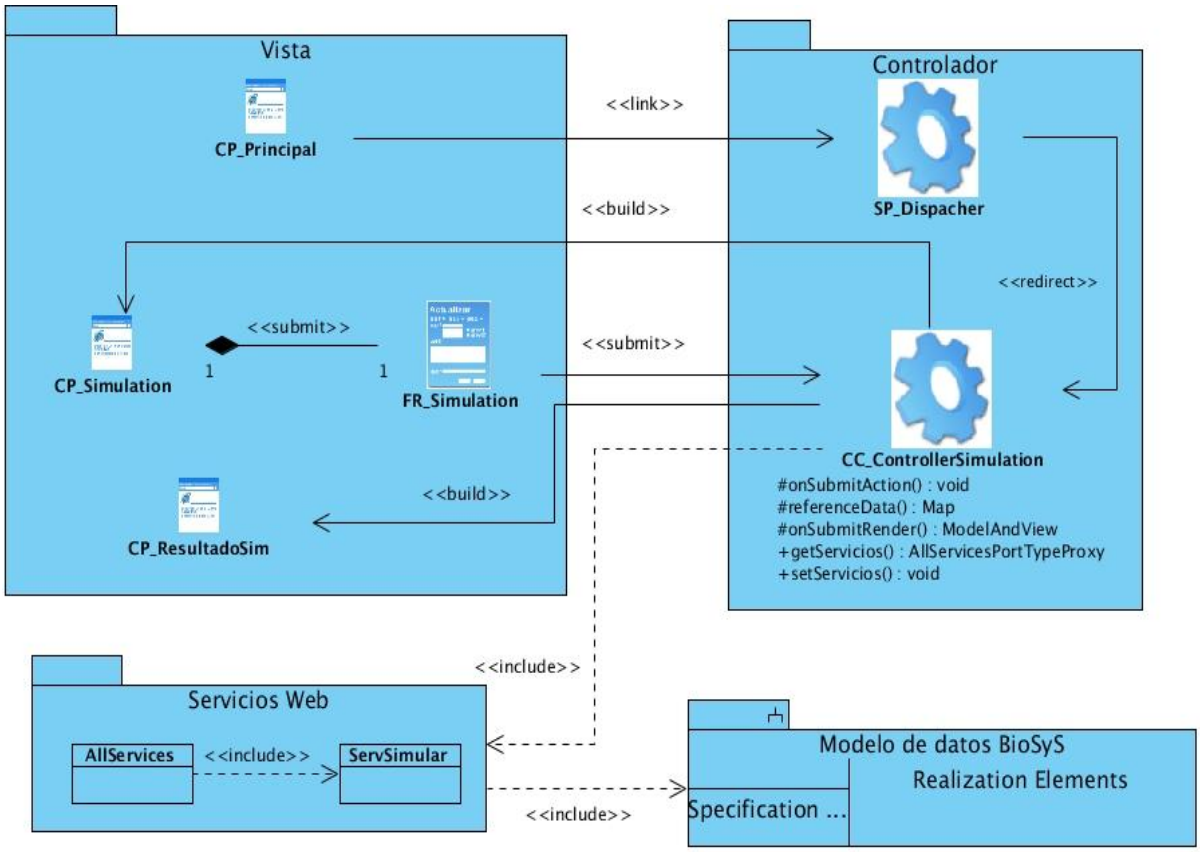


Fig. 8: Diagrama de Clases del Diseño del CU Simular. Se representan las páginas clientes (CP) agrupadas en el paquete Vista, las clases controladoras (CC) encapsuladas en el paquete Controlador, así como los paquetes destinados a gestionar los servicios web y a la base de datos.

Tabla 4: Descripción de las clases del diseño del CU Simular.

CP_Principal	Esta clase representa la vista principal a la cual tendrá acceso el usuario.
CP_Simulation	Clase dedicada a recoger los datos iniciales que ingresó el usuario para la simulación.
CP_ResultadoSim	Se diseñó para mostrar el resultado de la simulación al usuario.
SP_Dispacher	Clase controladora principal, maneja todos los eventos de la aplicación.

CC_ControllerSimulation	Concebida para interactuar con los datos ingresados en la página cliente “Simulation”.
AllServices	Clase destinada a invocar los servicios web.
ServSimular	Clase dedicada a la implementación del servicio web para simular.

3.6 Conclusiones

En el presente capítulo se detalló el uso del diagrama del diseño en el ciclo de vida de un software, específicamente en los CU Simular, posibilitando el entendimiento en el proceso de ejecución del caso de uso. Para el hacer el diagrama de secuencia se evidenció el uso de los patrones arquitectónicos y de diseño, los cuales proveen de mayor seguridad y robustez al producto final, posibilitando seleccionar una arquitectura orientada a servicios y cliente-servidor.

Para la implementación de la herramienta se utilizó el patrón arquitectónico Modelo-Vista-Controlador, el cual está presentes en los diagramas de clases del diseño. Los patrones de diseño utilizados fueron: Patrón Bajo Acoplamiento, Patrón Alta Cohesión, Patrón Creador, Patrón experto y Patrón controlador.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBAS

4.1 Introducción

En este capítulo, se expondrá el Diagrama de Componentes, donde se describen los elementos físicos del sistema y sus relaciones. Se muestran las clases e implementaciones más importantes y las pruebas realizadas para validar la solución.

4.2 Modelo de Implementación

El Modelo de Implementación describe cómo los elementos del Modelo de Diseño, se implementan en términos de Componentes, Ficheros de Código Fuente y Ejecutables. Describe también cómo se organizan los Componentes de acuerdo a los mecanismos de estructuración disponibles en el entorno de implementación y lenguaje o lenguajes de implementación empleados, y cómo dependen los Componentes unos de otros[55].

4.2.1 Diagrama de componentes

Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes de software: código fuente, binarios o ejecutables. Desde el punto de vista del diagrama de componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión del software, la reutilización, y las limitaciones imputadas por los lenguajes de programación y las herramientas utilizadas en el desarrollo.

A continuación se presenta el diagrama de componentes del CU Simular en la Figura 9. El resto de los diagramas de componentes, aparecen reflejados en el Documento de apoyo de la tesis.

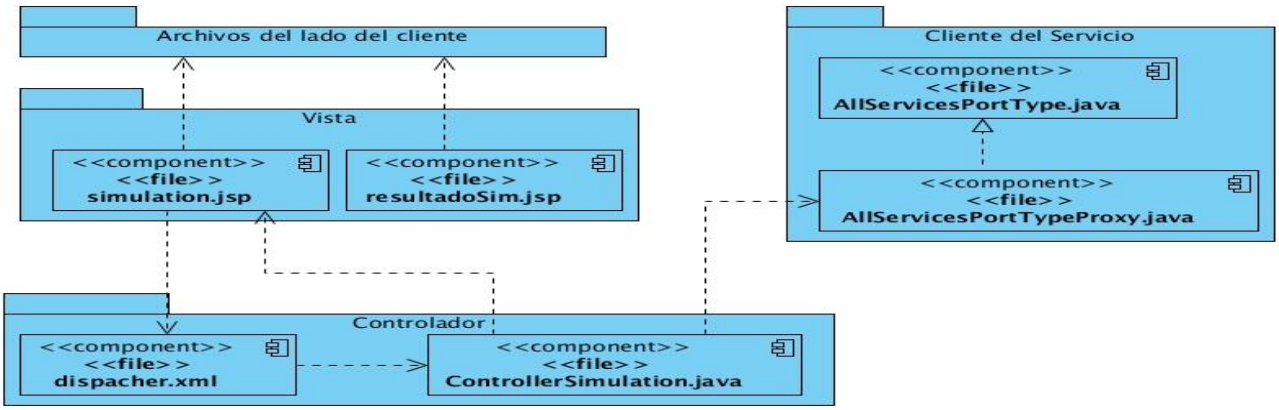


Fig. 9: Diagrama de componentes del CU Simular. Muestra la relación entre los diferentes componentes encapsulados en paquetes.

Tabla 5: Descripción de los componentes del diagrama del CU Simular.

Simulation.jsp	Página JSP que permite al usuario ingresar los datos para una simulación.
Dispacher.xml	Archivo XML encargado de configurar el controlador frontal, atender todas las peticiones del usuario y delegar estas al controlador indicado.
ResultadoSim.jsp	Página JSP utilizada para mostrar el resultado de la simulación.
ControllerSimulation.Java	Clase JAVA dedicada a procesar los datos enviados por la JSP.
AllServicesPortType.Java	Clase interfaz que declara todas las operaciones que invocan los servicios web.
AllServicesPortTypeProxy.Java	Clase que implementa las operaciones de la Interfaz AllServicesPortType.Java.

4.2.2 Diagrama de despliegue

Los diagramas de despliegue muestran a los nodos procesadores, la distribución de los procesos y de los componentes[54].

Los diagramas de despliegue (Figura 10) son los complementos de los diagramas de componentes que, unidos, proveen la vista de implementación del sistema. Describen la topología del sistema, la estructura de los elementos de hardware y el software que ejecuta cada uno de ellos. Los diagramas de despliegue representan a los nodos y sus relaciones. Los nodos son adheridos por asociaciones de comunicación tales como enlaces de red, conexiones TCP/IP, JDBC o RMI.

Los nodos representados, se conectan mediante soportes bidireccionales que pueden a su vez estereotiparse. 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, como instancias de componentes y objetos.

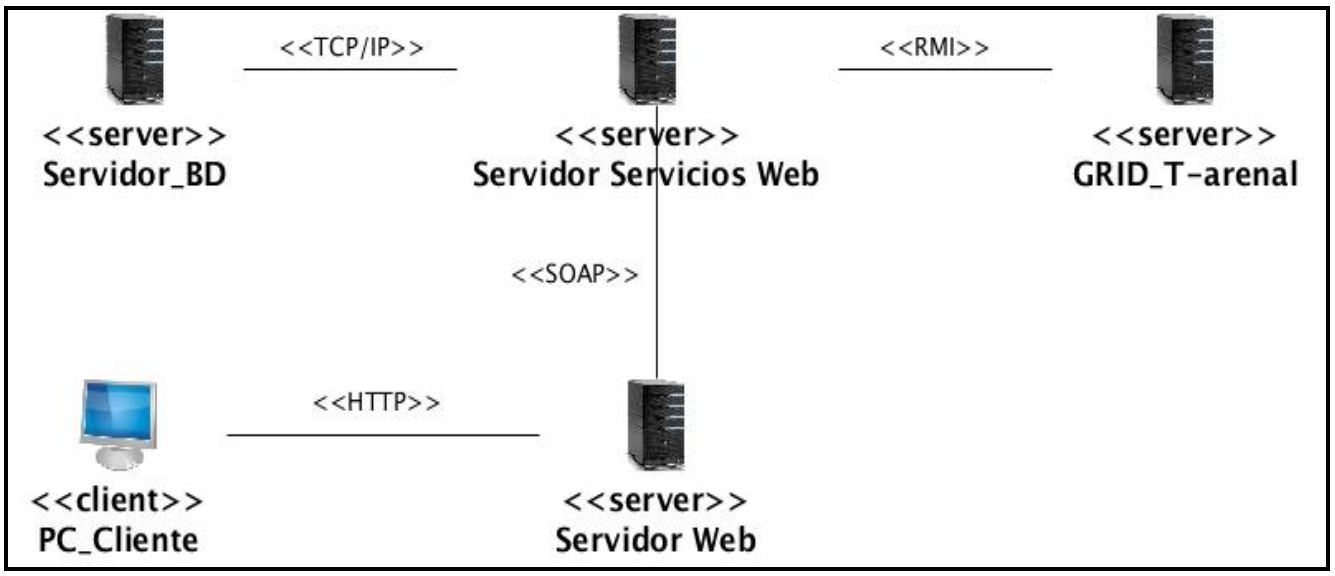


Fig. 10: Diagrama de despliegue de la aplicación. Se representan los componentes necesarios para realizar el despliegue del sistema y los protocolos de comunicación entre ellos.

PC_Cliente: Representa la máquina cliente destinada para el investigador y permite gestionar las peticiones del cliente. Se conecta a un servidor web por el protocolo HTTP.

Servidor_BD: Representa un servidor de base de datos destinado a guardar la información de los datos de la herramienta.

GRID_T-Arenal: Representa un servidor de objetos, recibe las peticiones del Servidor Servicios Web por el protocolo RMI, las divide en estaciones de trabajo y se la envía a sus máquinas clientes.

Servidor Servicios Web: Representa un servidor para recibir las peticiones del Servidor Web y envía la información al Servidor_BD y al servidor GRID_T-Arenal.

Servidor Web: Representa un servidor para hacer las peticiones de los servicios web al Servidor Servicios Web.

4.3 Implementaciones relevantes

A continuación se presentan algunos métodos significativos para el desarrollo de los servicios web que se utilizan en la solución del trabajo.

```
public String simular(BSOdeSetAlter ode, BSConditionsAlter[] bsvar,
        BSConditionsAlter[] bspar, String tool, double initT, double endT,
        int cantV, int cantP) {
    ArrayList<BSVariable> var = new ArrayList<BSVariable>();
    for (int i = 0; i < bsvar.length; i++) {
        BSVariable v = new BSVariable(bsvar[i].getBs_symbol(),
                bsvar[i].getBs_idCondition(), bsvar[i].getBs_values()[0],
                bsvar[i].getBs_idValues()[0]);
        v.setValues(bsvar[i].getValuesArray());
        v.setIdValues(bsvar[i].getIDValuesArray());
        var.add(v);
    }
    ArrayList<BSParameter> par = new ArrayList<BSParameter>();
    for (int i = 0; i < bspar.length; i++) {
        BSParameter p = new BSParameter(bspar[i].getBs_symbol(),
                bspar[i].getBs_idCondition(), bspar[i].getBs_values()[0],
                bspar[i].getBs_idValues()[0]);
        p.setValues(bspar[i].getValuesArray());
        p.setIdValues(bspar[i].getIDValuesArray());
        par.add(p);
    }
    BSOdeSet odeset = ode.getBSOdeSetNormal();
    String result = servicesSim.simularDistribuido(odeset, var, par, tool,
            initT, endT, cantV, cantP);
    return result;
}
```

Fig. 11: Implementación del método Simular en la clase AllServices. Esta clase está diseñada para invocar a todos los servicios web de la aplicación.

```
public String simularDistribuido(BSOdeSet odeSet,
        ArrayList<BSVariable> var, ArrayList<BSParameter> par, String tool,
        Double initT, Double endT, Integer cantV, Integer cantP) {
    try {
        createInput(odeSet, initT, endT, tool);
        File datos = new File(simulationfolder, "datos.dat");
        datos.deleteOnExit();
        ObjectOutputStream out = new ObjectOutputStream(
                new FileOutputStream(datos));
        ArrayList<BSCondition> cond = new ArrayList<BSCondition>();
        cond.addAll(var);
        cond.addAll(par);
        out.writeObject(cond);
        out.close();
        // TAREnAL
        UserAuthentication user = new UserAuthentication("yanier", "yanier");
        HostCommunication host = new HostCommunication("10.54.18.9", 6000);
        SystemConnection sys = new SystemConnection(Locale.getDefault(),
                user, host);
        // end TARENAL
        // subiendo problema
        Problem p = subirProblema(sys);
        System.out.println("Subio el problema con la descrpcion "
                + p.getDescription());
        // ejecutar el problema
        ejecutarProblema(odeSet, tool, p, sys, cantP, cantV);
    } catch (Exception e) {
        System.out.println("ERROR en simular ********** " + e.getMessage());
    }
    return "Ok pincha el metodo";
}
```

Fig. 12: Servicio Web para Simular. Está implementado en la clase ServSimular.

4.4 Prototipos funcionales de la aplicación

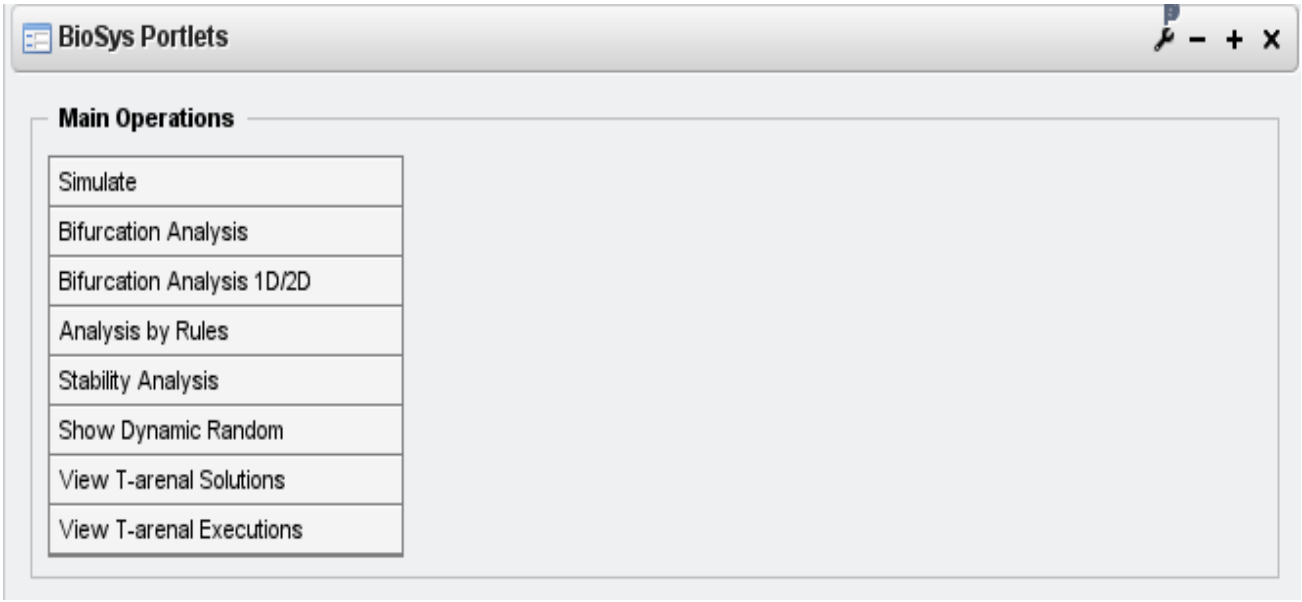


Fig. 13: Interfaz principal del portlet. Se presentan las opciones que tiene el investigador para realizar en la herramienta.

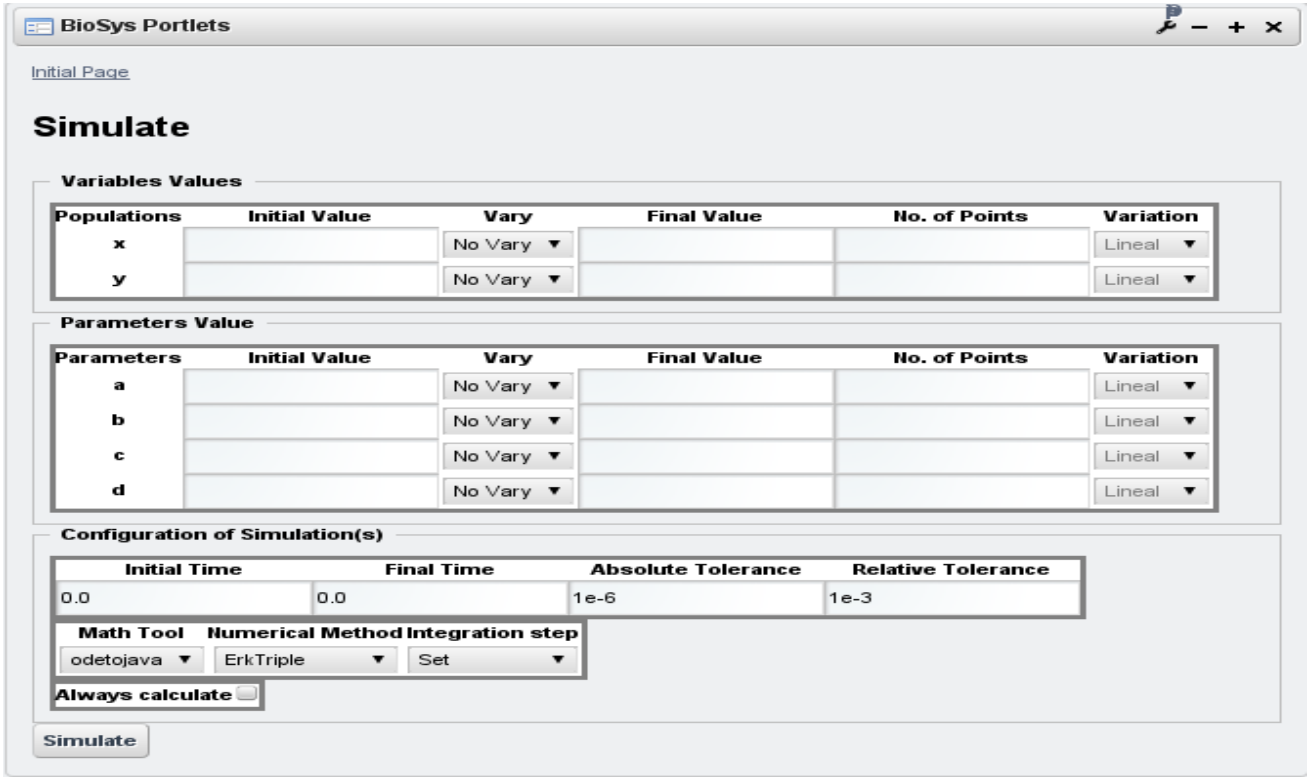


Fig. 14: Interfaz para simular un modelo matemático en la PSB. Presenta un formulario para ingresar los valores iniciales de las poblaciones y los parámetros, la variación del tiempo, así como la herramienta matemática a utilizar.

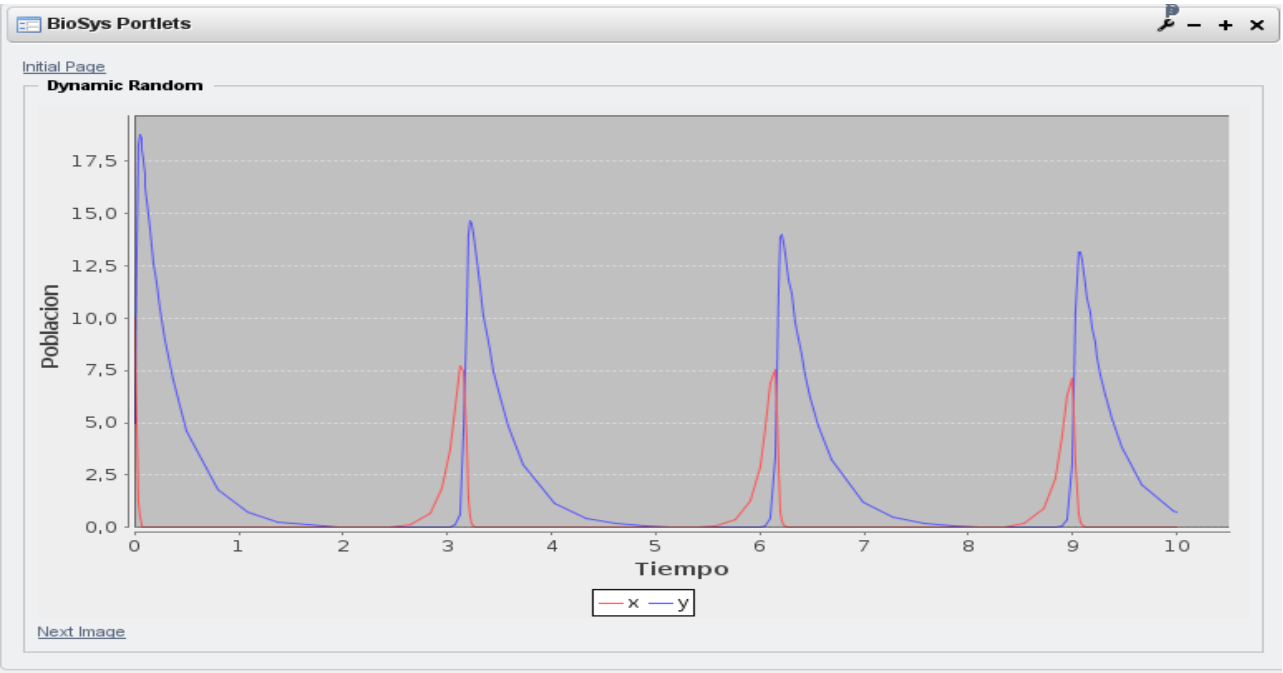


Fig. 15: Interfaz para visualizar las dinámicas aleatorias de las soluciones. Estos datos se obtienen de la base de datos.

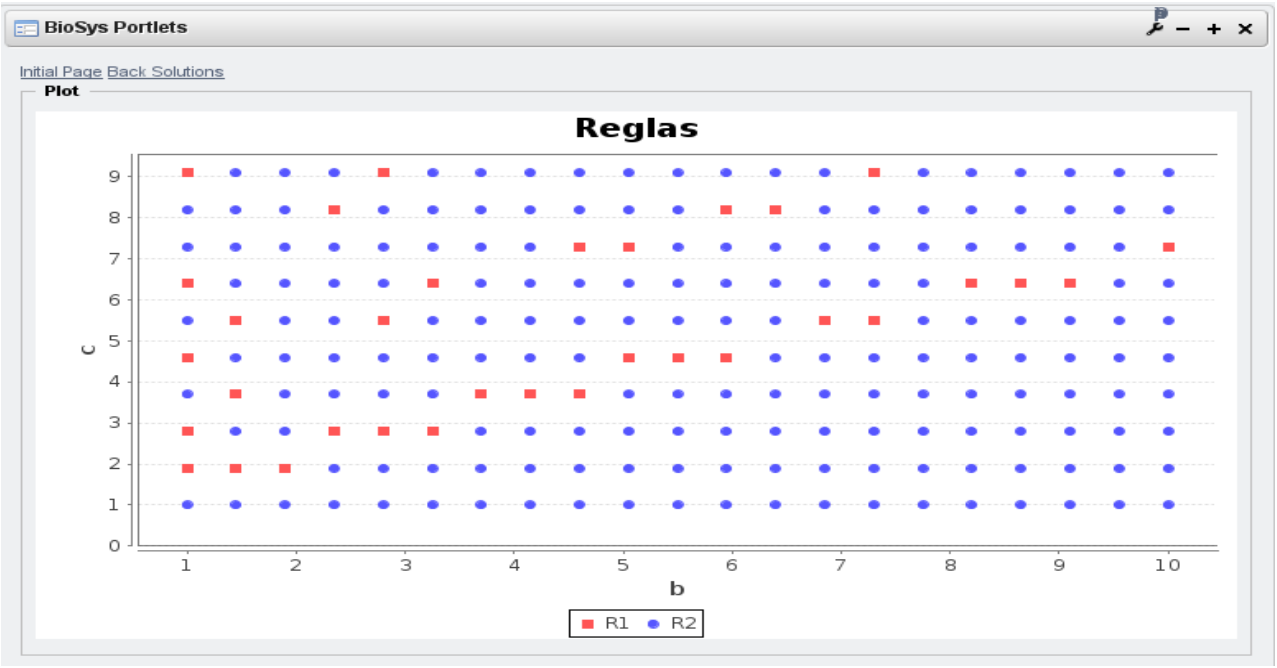


Fig. 16: Interfaz para mostrar la gráfica de análisis de bifurcaciones 1D/2D. Se puede acceder a la gráfica en las soluciones de T-Arenal.

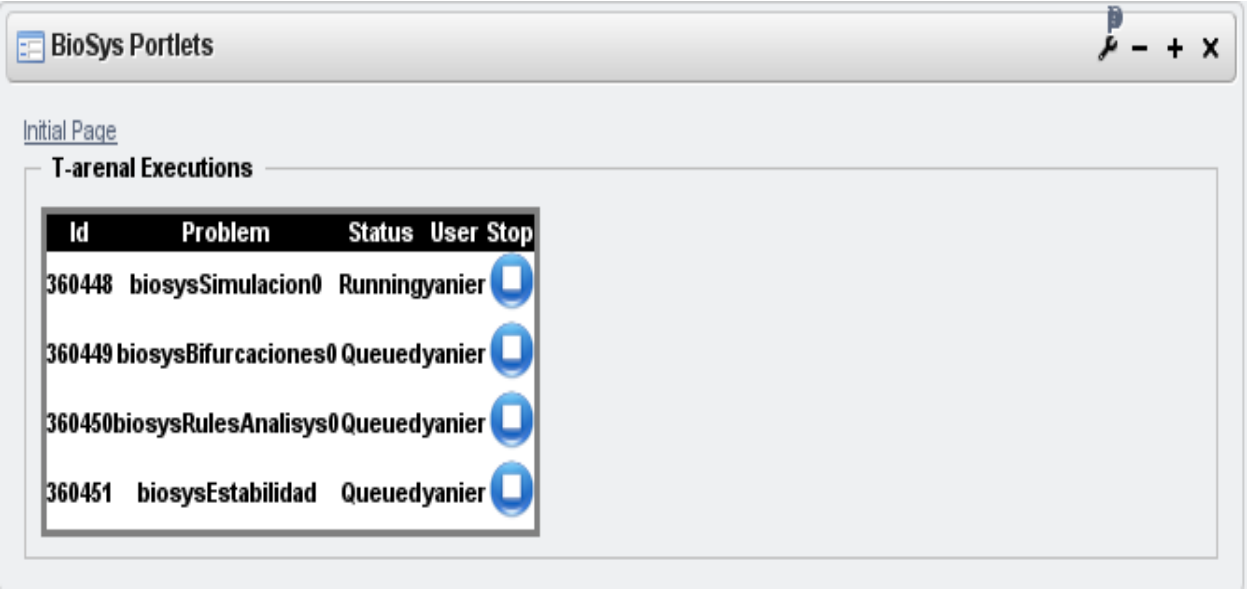


Fig. 17: Interfaz para mostrar las ejecuciones de T-Arenal. Se puede acceder desde la página principal y permite detener las ejecuciones de T-Arenal.

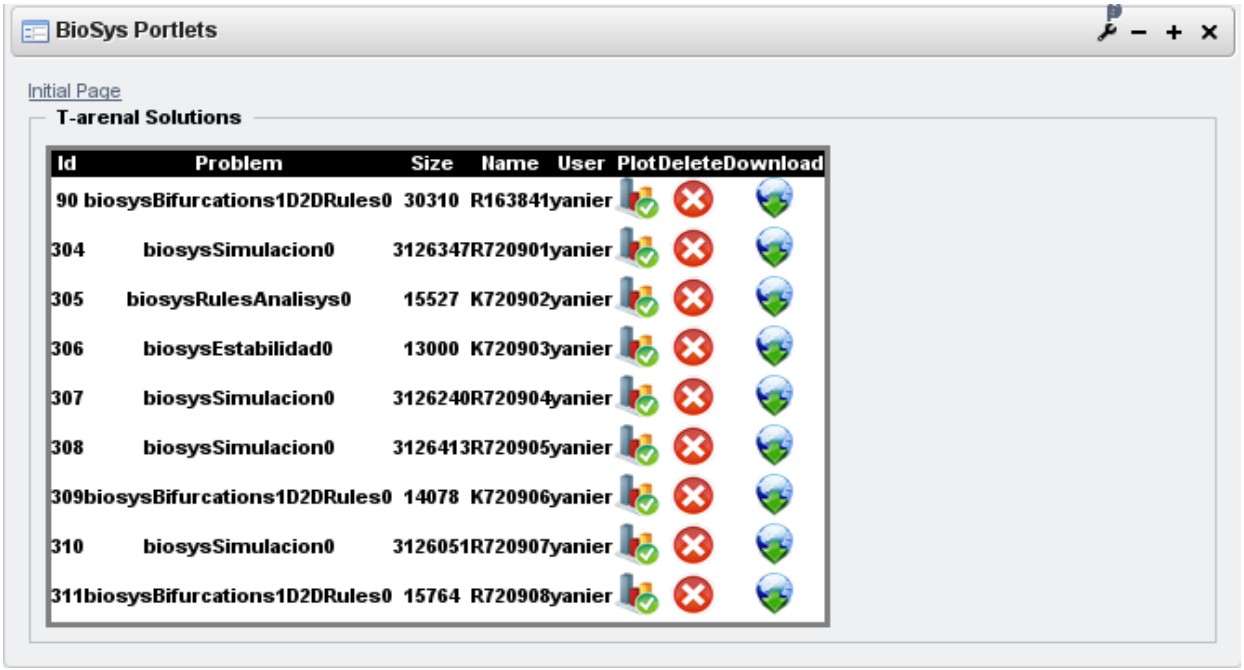


Fig. 18: Interfaz para mostrar las soluciones de T-Arenal. Permite eliminar y descargar las soluciones, así como ver el resultado de la gráfica de análisis de bifurcaciones 1D/2D.

4.5 Pruebas de software

La fase de pruebas es una de las más costosas del ciclo de vida software. En sentido estricto, deben realizarse pruebas de todos los artefactos generados durante la construcción de un producto, lo que

incluye especificaciones de requisitos, casos de uso, diagramas de diversos tipos, el código fuente y el resto de productos que forman parte de la aplicación[56]. Son esencialmente usadas para descubrir los errores en el software, tanto el código fuente como en la interfaz de usuario.

El éxito en las pruebas de software determina la satisfacción final del cliente, pruebas mal ejecutadas a una aplicación, puede generar daños económicos muy serios. Por eso esta fase es muy importante, se recomienda sea realizada por personas ajenas a la elaboración de la herramienta, lo cual subsistiría mayores dudas y por ende más pruebas al producto.
Los métodos para validar las aplicaciones informáticas en la UCI más utilizados son: Caja Negra y Caja Blanca.

4.5.1 Pruebas de Caja Negra

Las pruebas de caja negra, también denominadas pruebas de comportamiento, se centran en los requisitos funcionales del software. La prueba de caja negra intenta encontrar 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[57].

4.5.2 Pruebas de Caja Blanca

La prueba de caja blanca, denominada a veces prueba de caja de cristal es un método de diseño de casos de pruebas que usa la estructura de control del diseño procedimental para obtener los casos de prueba[58].

Mediante los métodos de prueba de caja blanca, el ingeniero del software puede obtener casos de prueba que: garanticen que se ejercita por lo menos una vez todos los caminos independientes de cada módulo; ejerciten todas las decisiones lógicas en sus vertientes verdadera y falsa; ejecuten todos los bucles en sus límites y con sus límites operacionales; y ejerciten las estructuras internas de datos para asegurar su validez.

4.5.3 Aplicación del método de Caja Negra

Para aplicar el método se obtuvieron las variables las cuales se definen en conjunto de entrada, condiciones de ejecución y resultados esperados al finalizar.

Tabla 6: Método Caja Negra al CU Simular.

Para mejor comprensión de la tabla se dividirá en secciones las variables, para que sean visibles todos los campos.

Escenario	Descripción	Initial Value	Vary	Final Value
EC Realizar Simulación con datos válidos.	Permite al investigador llenar los campos correctamente y enviarlos para realizar la simulación.	V: 2,0	V: vary	V: 3,5
EC Realizar Simulación con datos no válidos.	Error al intentar enviar datos para realizar la simulación.	I: -10	V: no vary	V: ( )

Escenario	No. of Points	Variation	Initial Time	Final Time	Absolute Tolerance
EC Realizar Simulación con datos válidos.	V: 2	V: log(x)	V: 0,0	V: 10,0	V: -1
EC Realizar Simulación con datos no válidos.	V: 3	V: lineal	V: 0,0	V: 10,0	V: -1

Escenario	Relative Tolerance	Math Tool	Numerical Method	Integration Step	No Point
EC Realizar Simulación con datos válidos.	V: -1	V: odetoJava	V: ErkTriple	V: Set	V: 2
EC Realizar Simulación con datos no válidos.	V: -1	V: odetoJava	V: ErkTriple	V: Set	V: 2

Escenario	Step	Initial Step Size	Max Step Size	Respuesta del sistema	Flujo central

EC Realizar Simulación con datos válidos.	V: 3	V: 1,0	V: 10	Realiza la simulación y muestra los resultados de dicha simulación.	El investigador selecciona con un clic la opción Simulate. Presiona clic en uno de los modelos disponibles. Se muestra una ventana donde se introducen los datos. Se presiona el botón Simulate y se muestra una ventana indicándole que se están simulando los datos.
EC Realizar Simulación con datos no válidos.	V: 3	V: 1,0	V: 10	Mu estra un mensaje de error en el campo no válido.	El investigador selecciona con un clic la opción Simulate. Presiona clic en uno de los modelos disponibles. Se muestra una ventana donde se introducen los datos. Se

					presiona el botón Simulate y se muestra un mensaje de error indicándole que hay valores no válidos.

Tabla 7: Descripción de las variables

No.	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Initial Value	Campo de Texto	No	El campo estará compuesto por un número. Las restricciones que se validan son: valor nulo, valor negativo y la inserción de caracteres especiales.
2	Vary	Lista desplegable	Si	El campo se podrá seleccionar o no, según decida el investigador. Si se selecciona toma valor: vary, en caso contrario toma valor: no vary por defecto.
3	Final Value	Campo de Texto	Si	El campo podrá ser nulo en caso de no haberse seleccionado anteriormente la variable Vary, en caso contrario estará compuesto por un número y las restricciones que se validan son: valor nulo, valor negativo y la inserción de caracteres especiales.

4	No. of Point	Campo de Texto	No	El campo estará compuesto por un número. Las restricciones que se validan son: valor nulo, valor negativo y la inserción de caracteres especiales.
5	Variation	Lista desplegable	No	El campo estará compuesto por una cadena de caracteres. Si el campo Vary fue seleccionado los posibles valores a seleccionarse son: lineal, log(x), ln(x), exp(x). En caso contrario toma valor: lineal por defecto.
6	Initial Time	Campo de Texto	No	El campo estará compuesto por un número. Las restricciones que se validan son: valor nulo, valor negativo y la inserción de caracteres especiales.
7	Final Time	Campo de Texto	No	El campo estará compuesto por un número. Las restricciones que se validan son: valor nulo, valor negativo y la inserción de caracteres especiales.
8	Absolute Tolerance	Campo de Texto	No	El campo estará compuesto un número. Las restricciones que se validan son valor nulo y la inserción de caracteres especiales.

9	Relative Tolerance	Campo de Texto	No	El campo estará compuesto un número. Las restricciones que se validan son valor nulo y la inserción de caracteres especiales.
10	Math Tool	Lista desplegable	No	El campo estará compuesto por un listado de herramientas matemáticas, de este listado el investigador selecciona una.
11	Numerical Method	Lista desplegable	No	El campo estará compuesto por un listado de métodos numéricos, de este listado el investigador selecciona uno.
12	Integration Step	Lista desplegable	No	El campo estará compuesto por un listado de pasos de integración, de este listado el investigador selecciona uno.
13	No Points	Campo de Texto	No	El campo estará compuesto por un número. Las restricciones que se validan son: valor nulo, valor negativo y la inserción de caracteres especiales.
14	Step	Campo de Texto	No	El campo estará compuesto por un número. Las restricciones que se validan son: valor nulo, valor negativo y la inserción de caracteres especiales.

15	Initial Step Size	Campo de Texto	No	El campo estará compuesto por un número. Las restricciones que se validan son: valor nulo, valor negativo y la inserción de caracteres especiales.
16	Max Step Size	Campo de Texto	No	El campo estará compuesto por un número. Las restricciones que se validan son: valor nulo, valor negativo y la inserción de caracteres especiales.

4.5.4 Aplicación del método de Caja Blanca

Camino básico

Para la ejecución de esta técnica existen cuatro pasos básicos:

- A partir del diseño o del código fuente, se dibuja el grafo de flujo asociado.
- Se calcula la complejidad ciclomática del grafo.
- Se determina un conjunto básico de caminos independientes.
- Se preparan los casos de prueba que obliguen a la ejecución de cada camino del conjunto básico.

Camino Básico. Clase ServSimular. Método ejecutarProblema.

```
private void ejecutarProblema(BSOdeSet odeSet,String herramienta, Problem p, SystemConnection sys, Integer ce
1   try {
2       File[] externalFiles;
2       File f;
2       ModeloNew modelo = getListModel(getListSystem().get(0).getId()).get(0);
3 y 4   if (herramienta.equalsIgnoreCase("matlab") || herramienta.equalsIgnoreCase("octave")) {
5           if (herramienta.equalsIgnoreCase("octave")) {
6               f = new File(simulationfolder, modelo.getName() + "1.m");
6               BufferedWriter bw = new BufferedWriter(new FileWriter(f));
6               bw.write(modelo.getMatlabmodel());
6               bw.close();
6               CrearOctaveModelFromMatlLabModel(f, cantP, cantV, modelo.getName(), odeSet.getIntegration_met
6               f.delete();
6               f = new File(simulationfolder, modelo.getName() + ".m");
7           } else {
7               f = new File(simulationfolder, modelo.getName() + ".m");
7               BufferedWriter bw = new BufferedWriter(new FileWriter(f));
7               bw.write(modelo.getMatlabmodel());
7               bw.close();
            }
8               externalFiles = new File[3];
8               externalFiles[0] = new File(simulationfolder, "datos.dat");
8               externalFiles[1] = new File(simulationfolder, "input.xml");
8               externalFiles[2] = f;
9       } else {
9           f = new File(simulationfolder, modelo.getName() + ".java");
9           BufferedWriter bw = new BufferedWriter(new FileWriter(f));
9           bw.write(modelo.getJavamodel());
9           bw.close();
9           String[] optionsAndSources = {
                "-g", "-source", "1.5", "-target", "1.5", "-cp", "grid/lib/BSSimulacion.jar", f.toString()
            };
9           PrintWriter out = new PrintWriter(new FileWriter(new File(f.getParent(), "out.txt")));
9           Main.compile(optionsAndSources, out);
9           f.delete();
9           f = new File(simulationfolder, modelo.getName() + ".class");
9           f.deleteOnExit();
9           externalFiles = new File[3];
9           externalFiles[0] = new File(simulationfolder, "datos.dat");
9           externalFiles[1] = new File(simulationfolder, "input.xml");
9           externalFiles[2] = f;
        }
10      ExecutionFilesUploader job = sys.getProblemManager().executeProblem(p.getKeyAccess(), externalFiles);
10      job.start();
11  } catch (Exception e) {
        // TODO: handle exception
    }
}
```

Fig. 19: Técnica Camino Básico. Código fuente. Clase: ServSimular. Método: ejecutarProblema. Paso #1.

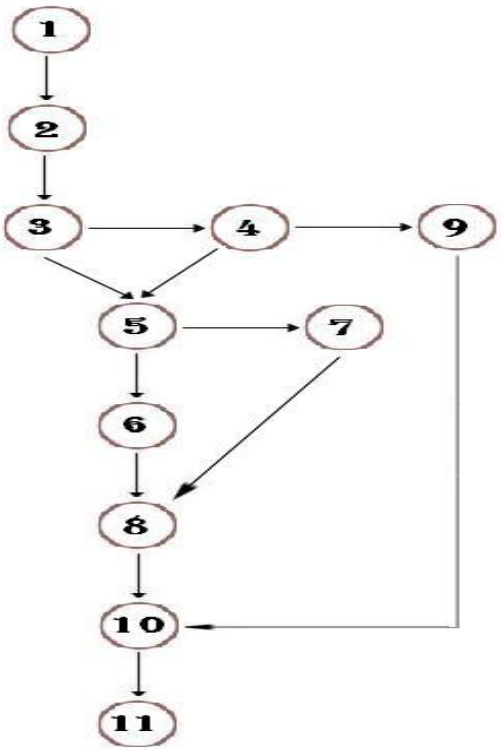


Fig. 20: Técnica Camino Básico. Ciclo de ejecución. Clase: ServSimular. Método: ejecutarProblema. Paso #1.

Paso 2: Complejidad Ciclomática.

Proporciona una medición cuantitativa de la complejidad lógica de un programa, define el número de caminos independientes del conjunto básico de este y brinda el límite superior para el número de pruebas a realizar que aseguren la ejecución de todas las sentencias al menos una vez. La Complejidad Ciclomática se denota por V y una de las vías utilizadas para calcularla es V (G) = P + 1. Donde, P es el número de nodos predicado (nodo de donde salen más de una arista) del grafo G.

V (G) = P + 1
V (G) = 3 + 1
V (G) = 4

Paso 3: Determinar un conjunto de Caminos Básicos Independientes
Camino #1: 1 – 2 – 3 – 4 – 9 – 10 – 11.
Camino #2: 1 – 2 – 3 – 5 – 6 – 8 – 10 – 11.
Camino #3: 1 – 2 – 3 – 4 – 5 – 6 – 8 – 10 –11.
Camino #4: 1 – 2 – 3 – 4 – 5 – 7 – 8 – 10 – 11.

Camino Básico. Clase ServSimular. Método CrearOctaveModelFromMatLabModel.

```
private void CrearOctaveModelFromMatlLabModel(File MatLabModel, Integer CantP, Integer CantEcua, String name, String

1   try {
2       BufferedWriter bwLsode = new BufferedWriter(new FileWriter(new File(MatLabModel.getParentFile(), name + ".m")

2       BufferedReader br = null;
2       FileReader fr = null;
2       fr = new FileReader(MatLabModel);
2       br = new BufferedReader(fr);
2       String linea;
3       while ((linea = br.readLine()) != null) {
4,5,6       if (!linea.startsWith("y_y") && !linea.startsWith("p_p") && !linea.startsWith("if")) {
7               bwLsode.write(this.GetLineaActual(linea, name, metodo) + "\n");

8               if (linea.startsWith("%PR")) {
8                   bwLsode.write("%EC " + CantEcua + "\n");
                }
9               if (linea.startsWith("function")) {
10                  for (int i = 1; i <= CantEcua; i++) {
10                      bwLsode.write(InicializarY_Y(i));
                    }
11                  for (int i = 1; i <= CantP; i++) {
11                      bwLsode.write(InicializarP_P(CantEcua, i));
                    }
                }
12              if (linea.startsWith("error")) {
13                  for (int j = 1; j <= (CantP + CantEcua - CantEcua); j++) {
13                      bwLsode.write("r_r(" + (j + CantEcua) + ")=0;\n");
                    }
                }

            }
        }
14      bwLsode.close();
15  } catch (Exception e) {
        // TODO: handle exception
    }

}
```

Fig. 21: Técnica Camino Básico. Código fuente. Clase: ServSimular. Método: CrearOctaveModelFromMatLabModel. Paso #1.

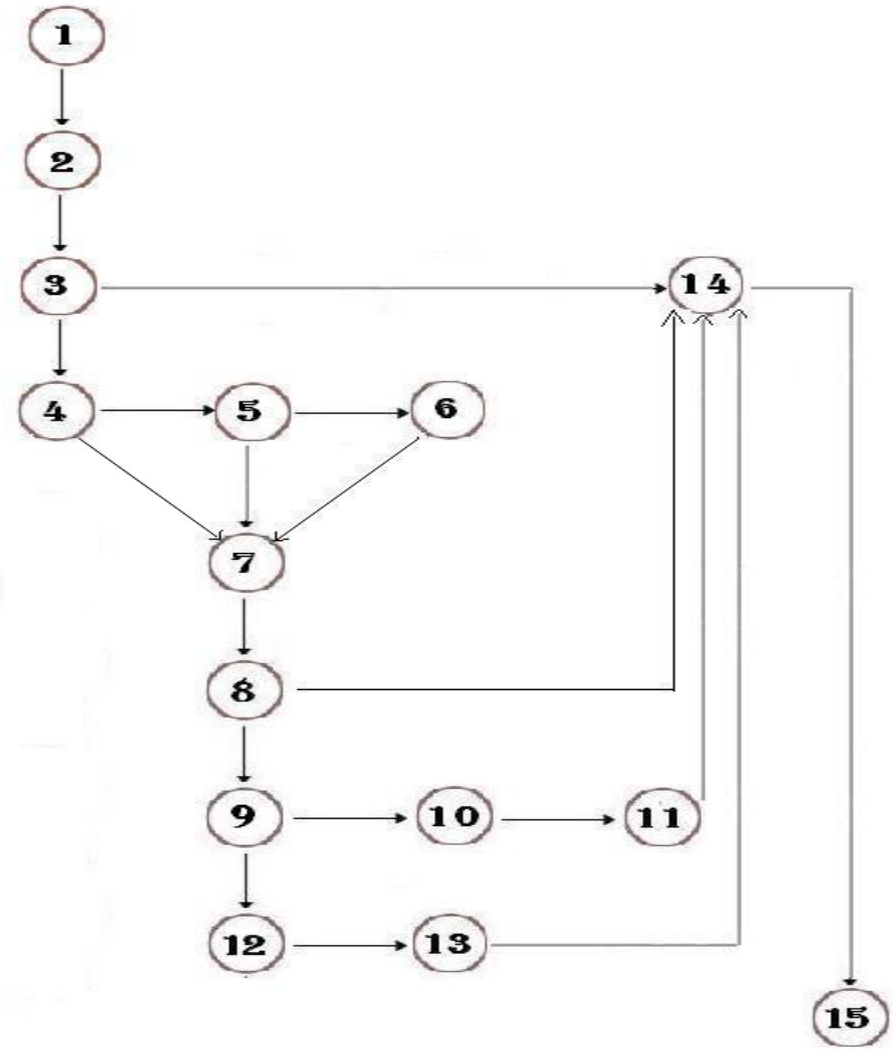


Fig. 22: Técnica Camino Básico. Ciclo de ejecución. Clase: ServSimular. Método: CrearOctaveModelFromMatLabModel. Paso #1.

Paso 2: Complejidad Ciclomática.

V (G) = P + 1

V (G) = 5 + 1

V (G) = 6

Paso 3: Determinar un conjunto de Caminos Básicos Independientes.

Camino #1: 1 – 2 – 3 – 14 – 15.

Camino #2: 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9– 10 – 11– 14 – 15.

Camino #3: 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 12 – 13– 14 – 15.

4.5 Conclusiones

En el presente capítulo se creó el modelo de implementación, que arrojó como resultado el diagrama de componentes del CU Simular, posibilitando un acercamiento a los componentes más importantes para el funcionamiento del sistema, encapsulados en paquetes para una mejor comprensión. Se realizó el diagrama de despliegue, representando los nodos físicos necesarios para la ejecución de la herramienta computacional.

Se presentaron algunas implementaciones significativas, perteneciente al servicio web para Simular. Se mostraron además, los prototipos funcionales del sistema, o sea las interfaces del portlet.

Se definieron los conceptos de Pruebas de Caja Negra y Pruebas de Caja Blanca, las cuales permitieron validar las funcionalidades implementadas en los servicios web, así como en el portlet. Las Pruebas de Caja Negra se realizaron sobre la interfaz de usuario, y se detectaron un grupo de no conformidades las cuales arrojaron errores como parámetros nulos, no validación de campos, excepciones no tratadas y ficheros vacíos, las cuales fueron corregidas. Las pruebas de Caja Blanca se le realizaron al código para comprobar los caminos lógicos del software, proponiéndose Casos de Prueba donde se ejercitaron conjuntos específicos de condiciones, las cuales garantizaron la ejecución de cada sentencia del programa al menos una vez durante las pruebas.

CONCLUSIONES

Finalizada la investigación del trabajo se desarrollaron los servicios web de las funcionalidades simulación y análisis para la Plataforma de Servicios Bioinformáticos.

Se desarrolló un portlet, el mismo consume los servicios web implementados del Módulo Simulación y el Módulo Análisis. Podrá ser usado de forma inherente por cualquier tecnología ya que posee código HTML, el cual es muy común para los desarrolladores de aplicaciones web.

Se realizó una validación en las funcionalidades de la herramienta, mediante el método de Caja Negra, evidenciándose errores en la interfaz de usuarios, que fueron corregidos para eliminar las no conformidades. Mediante el método de Caja Blanca se le realizaron pruebas al código para comprobar los caminos lógicos del software, donde se ejercitaron conjuntos específicos de condiciones, las cuales garantizaron la ejecución de cada sentencia del programa al menos una vez durante las pruebas.

RECOMENDACIONES

- Incorporar las funcionalidades del Módulo para la Modelación Gráfica a la solución propuesta en el trabajo.
- Incorporar las funcionalidades del Módulo Editor de Ecuaciones a la solución propuesta en el trabajo.

BIBLIOGRAFÍA

[1] Sánchez, D; Hansen, H. Implementación de las Tecnologías Web (Internet) para la integración, fluidez y la optimización de los procesos académicos en la dirección docente de la Universidad del Zulia. [En línea], 2007. [Consultado: mayo de 2012]. Disponible en: [http://www.urbe.edu/publicaciones/telematica/indice/pdf-vol6-3/7-implementacion-de-las-tecnologías].

[2] Martínez Usero, J. Á. Análisis de los usuarios, contenidos y servicios de los servicios públicos electrónicos. [En línea], 2007. [Consultado: abril de 2012]. Disponible en: [http://eprints.ucm.es/6253/1/2007-BAAB-administracion.pdf].

[3] Cañedo, R.; Arencibia, R. Bioinformática: en busca de los secretos moleculares de la vida. [En línea], Diciembre 2004. [Consultado: diciembre de 2011], Disponible en: [http://scielo.sld.cu/pdf/aci/v12n6/aci02604.pdf].

[4] Moreno, Noel. BioSyS: Software para la simulación y análisis de sistemas. La Habana, Cuba, 2007. 93.

[5] Pérez Capdevila, J. Las Tecnologías Web para la Gestión del Conocimiento. [En línea], Septiembre 2007. [Consultado: enero de 2012]. Disponible en.: [http://www.sociedadelainformacion.com/9/las_tecnologias_web.htm].

[6] Voos, J. A.; Ing. González, E.; Ing. Cagnolo, F. Portal de Aplicaciones Médicas. [En línea]. 2007. [Consultado: febrero de 2012]. Disponible en: [http://www.bioingenieria.edu.ar/grupos/geic/biblioteca/Trabypres/T03TCAr10.pdf].

[7] García Gómez, J. C. Portal: definición, evolución y clasificación. Murcia, España. [En línea]. 2003. [Consultado: marzo de 2012]. Disponible en: [http://aprendeenlinea.udea.edu.co/lms/moodle/file.php/87/documentos/definicion-portales.pdf].

[8] DNA Data Bank of Japan (DDBJ). [En línea], 2010. [Consultado: Abril 2012]. Disponible en: [http://www.ddbj.nig.ac.jp/intro-e.html].

[9] National Center for Biotechnology Information (NCBI). [En línea], 2004. [Consultado: Abril 2012]. Disponible en: [http://www.ncbi.nlm.nih.gov/About/glance/ourmission.html].

[10] Bioinformatics Resource Portal (Expasy). [En línea], 2011. [Consultado: Abril 2012]. Disponible en: [http://expasy.org].

[11] European Bioinformatics Institute (EBI). [En línea], 2011. [Consultado: Abril 2012]. Disponible en: [http://www.ebi.ac.uk/Information].

[12] Agramonte, A.; Martínez, O. *UCIBIOSOFT: Portal Web de Servicios Bioinformáticos.* La Habana, Cuba, 2011.

[13] Arboleda, L. M. Servicios web: distribución e integración. [En línea]. Octubre 2004. [Consultado: mayo de 2012]. Disponible en: [http://www.icesi.edu.co/biblioteca_digital/bitstream/10906/403/1/larboled_servicios-web.pdf].

[14] Cuellar, I. Plataforma de Aprendizaje a Distancia Basada en tecnología de Portlets y Web 2.0 para una Enseñanza Participativa. [En línea]. 2008. [Consultado: abril de 2012]. Disponible en: [http//eprints.ucm.es90831memoriaSI_0708.pdf].

[15] Arias, G. *Editor de ecuaciones para la Plataforma de Simulación de Sistemas Biológicos.* La Habana, Cuba, 2008. 73

[16] Rational Unified Process. Best Practices for Software Development Teams. [En línea]. 2005. [Consultado: junio de 2012]. Disponible en: [http://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf].

[17] Extreme Programming. [En línea]. 2011. [Consultado: Mayo 2012]. Disponible en: [http://www.extremeprogramming.org].

[18] Torres Flores, C. L.; Alférez Salinas, G. H. Establecimiento de una Metodología de Desarrollo de Software para la Universidad de Navojoa usando OpenUP. [En línea]. 2008. [Consultado: mayo de 2012] Disponible en: [http://fit.um.edu.mx/CIDET/publicaciones/COMP-004-2008_Establecimiento_de_una_Metodología_de_Desarrollo_de_Software_para_la_Universidad_de_Navojoa_Usando_OpenUP.pdf].

[19] Jacobson, I.; Booch, G.; Rumbaugh J. El proceso unificado de desarrollo de software. México. Adison Wesley. 2000. 458.

[20] Instituto Nacional de Estadística e Informática. Herramientas CASE. [En línea]. Noviembre 1999. [Consultado: junio de 2012]. Disponible en: [http://www.inei.gob.pe/biblioineipub/bancopub/Inf/Lib5103/Libro.pdf].

[21] Valdiviezo Basauri, M. J. Análisis y Diseño de una herramienta de desarrollo de soluciones para inteligencia de negocios-Análisis dimensional. [En línea]. . 2007. [Consultado: abril de 2012]. Disponible en: [http://tesis.pucp.edu.pe/repositorio/bitstream/handle/123456789/327/VALDIVIEZO_MANUEL_AN%C3%81LISIS_Y_DISE%C3%91O_DE_UNA_HERRAMIENTA_DE_DESARROLLO_DE_SOLUCIONES_PARA_INTELIGENCIA_DE_NEGOCIOS_AN%C3%81LISIS_DIMENSIONAL.pdf?sequence=1].

[22] Grupo Soluciones Innova. [En línea]. [Consultado: Junio 2012]. Disponible en: [http://www.rational.com.ar/herramientas/roseenterprise.html]

[23] Valade, J. PHP 5 for dummies. Wiley Publishing, Inc. Indianapolis, Indiana. 2004. 411.

[24] Cox, K. ASP.NET for dummies. Wiley Publishing, Inc. Indianapolis, Indiana. 2008. 435.

[25] Belmonte Fernández, O. Introducción al lenguaje de programación Java. Una guía básica. [En línea]. 2005. [Consultado: febrero de 2012]. Disponible en: [http://www3.uji.es/~belfern/pdidoc/IX26/Documentos/introJava.pdf].

[26] Mozilla Developer Network. [En línea]. [Consultado: Mayo 2012]. Disponible en: [https://developer.mozilla.org/es/Guia_JavaScript_1.5/Concepto_de_JavaScript].

[27] Eclipse Foundation. [En línea] . [Consultado: Mayo 2012]. Disponible en: [http://www.eclipse.org].

[28] Apache Portals. [En línea]. [Consultado: Junio 2012]. Disponible en: [http://portals.apache.org/jetspeed-1].

[29] JBoss Portal. [En línea]. [Consultado: Junio 2012]. Disponible en: [http://www.jboss.org/jbossportal].

[30] Yuan, J. Liferay Portal Enterprise Intranets. Birmingham, UK. Packet Publishing Ltd. 2008. 408.

[31] Durán, A.; Medel, R. Introducción a Apache Tomcat 5.5. [En línea]. Febrero 2007. [Consultado: marzo de 2012] Disponible en: [http://www.lsi.us.es/docencia/get.php?id=1923].

[32] Mateu, C. Desarrollo de aplicaciones web. Barcelona, España. [En línea]. Marzo 2004. [Consultado : abril 2012]. Disponible en: [http://books.openlibra.com/pdf/desarrollo_aplicaciones_web.pdf].

[33] Apache Software Foundation. [En línea]. 2010 [Consultado: Mayo de 2012]. Disponible en: [http://axis.apache.org/axis2/c/core].

[34] Goodwill, J. Mastering Jakarta Struts. Wiley Publishing, Inc., Indianapolis, Indiana. 2002. 262.

[35] Java Server Faces. [En línea]. 2010. [Consultado: Junio 2012]. Disponible en: [http://www.Javaserverfaces.org].

[36] Oracle. Hardware and Software Engineered to Work Together. [En línea]. 2010. [Consultado: Junio de 2012]. Disponible en: [http://www.oracle.com/technetwork/Java/Javaee/Javaserverfaces-139869.html].

[37] Fernández, N. Introducción a Spring. [En línea]. 2009. [Consultado: mayo de 2012]. Disponible en: [http://www.it.uc3m.es/mario/si/Introduccion_a_Spring.pdf].

[38] Raible, M. Comparing Java Web Frameworks. [En línea]. Mayo 2007. [Consultado: junio de 2012]. Disponible en: [http://static.raibledesigns.com/repository/presentations/ComparingJavaWebFrameworks-ApacheConUS2007.pdf].

[39] Portal en español sobre PostgreSQL. [En línea]. 2010. [Consultado: Mayo de 2012]. Disponible en:

[http://www.postgresql.org.es/sobre_postgresql].

[40] Larman, C.. UML y Patrones. Introducción al análisis y diseño orientado a objetos. México. Prentice Hall. 1999. 499.

[41] Martínez Jera, E; González Enríquez, L. AlasClínicas: Desarrollo de la Gestión de Datos de Ensayos Clínicos a partir del sistema OpenClinica. La Habana, Cuba. 2009.

[42] Presmman, R. Ingeniería de software. Un enfoque prático. McGraw-Hill. Estados Unidos. 2005. 900.

[43] Camacho, E; Cardeso, F; Nuñez, G. Arquitecturas de Software. [En línea]. Abril 2004. [Consultado: mayo de 2012]. Disponible en: [http://prof.usb.ve/lmendoza/Documentos/PS-6116/Guia_Arquitectura.pdf].

[44] Modelo Vista Controlador. Definición y características. [En línea]. Noviembre 2010. [Consultado: mayo de 2012]. Disponible en: [http://www.comusoft.com/modelo-vista-controlador-definicion-y-caracteristicas].

[45] Campo, G. Patrones de Diseño, Refactorización y Antipatrones. Ventajas y Desventajas de su Utilización en el Software Orientado a Objetos. [En línea]. (2009). [Consultado: mayo de 2012]. Disponible en: http://www.ucasal.net/templates/unid-academicas/ingenieria/apps/4-p101-Campo.pdf.

[46] Trott, J.; Shalloway, A. Design Patterns Explained. MODELER. Lugar de publicación desconocido. 2004. 334.

[47] Visconti, M; Astudillo, H. Fundamentos de Ingeniería de Software. [En línea]. 2005. [Consultado: abril de 2012]. Disponible en: [http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf].

[48] Mendoza, J. Diseño del sistema de tarjeta de crédito con UML. En línea. 2007. [Consultado: mayo 2012]. Disponible en: [http://sisbib.unmsm.edu.pe/bibvirtualdata/Tesis/Basic/mendoza_nj/Cap5.pdf].

[49] Tello, J. Diagramas de Secuencia. [En línea]. 2007. [Consultado: abril de 2012]. Disponible en: [http://www2.uah.es/jcaceres/capsulas/DiagramaSecuencia.pdf].

[50] Jacobson, I.; Booch, G.; Rumbaugh J. *El proceso unificado de desarrollo de software*. Adison Wesley. México. 2000. 458.

[51] Usaola, M. Mantenimiento Avanzado de Sistemas de Información. [En línea]: 2008. [Consultado: 2005]. Disponible en: [http://alarcos.esi.uclm.es/doc/masi/doc/lec/parte5/polo-apuntesp5.pdf].

[52] Visconti, M; Astudillo, H. Fundamentos de Ingeniería de Software. [En línea]. 2005. [Consultado: abril de 2012]. Disponible en: [http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf].

[53] Mendoza, J. Diseño del sistema de tarjeta de crédito con UML. En línea. 2007. [Consultado: mayo 2012]. Disponible en: [http://sisbib.unmsm.edu.pe/bibvirtualdata/Tesis/Basic/mendoza_nj/Cap5.pdf].

[54] Tello, J. Diagramas de Secuencia. [En línea]. 2007. [Consultado: abril de 2012]. Disponible en: [http://www2.uah.es/jcaceres/capsulas/DiagramaSecuencia.pdf].

[55] Mendoza, J. Diseño del sistema de tarjeta de crédito con UML. En línea. 2007. [Consultado: mayo 2012]. Disponible en: [http://sisbib.unmsm.edu.pe/bibvirtualdata/Tesis/Basic/mendoza_nj/Cap5.pdf].

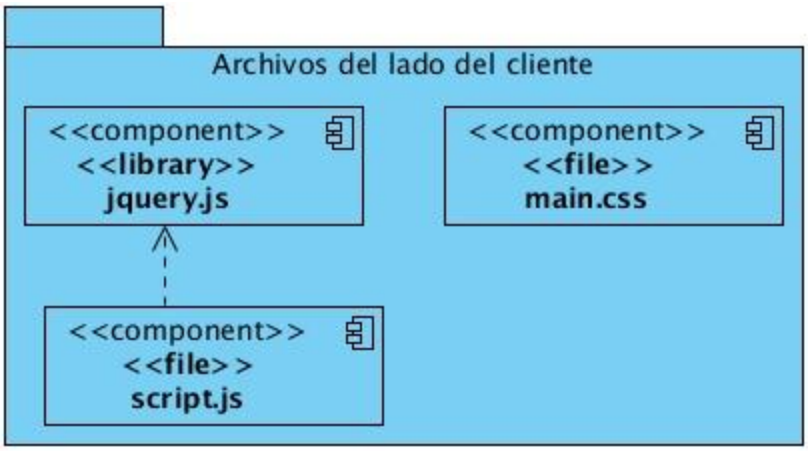
[56 Scribd. [En línea]. 2004. [Consultado: mayo de 2012]. Disponible en: [http://es.scribd.com/doc/53520152/9/Workflow-de-Implementacion].

[57] Larman, C.. UML y Patrones. Introducción al análisis y diseño orientado a objetos. México. Prentice Hall. 1999. 499.

[58] Jacobson, I.; Booch, G.; Rumbaugh J. *El proceso unificado de desarrollo de software*. Adison Wesley. México. 2000. 458.

ANEXOS

Anexo 1:1 Paquete de clases para el uso del investigador en la interfaz web.



Anexo 2: Vista básica de un portlet.

GLOSARIO

CU (Casos de Uso): es una técnica para la captura de requisitos potenciales de un nuevo Sistema.

EJB (Enterprise JavaBeans): son una de las API que forman parte del estándar de construcción de aplicaciones empresariales J2EE. Su especificación detalla cómo los servidores de aplicaciones proveen objetos desde el lado del servidor que son, precisamente, los EJB.

JVM (Java Virtual Machine): es una máquina virtual de proceso nativo, o sea se ejecuta en una plataforma específica, capaz de interpretar y ejecutar instrucciones expresadas en un código binario especial (el "bytecode" Java), el cual es generado por el compilador del lenguaje Java.

RMI (Java Remote Method Invocation): es un mecanismo ofrecido por Java para invocar un método de manera remota. Forma parte del entorno estándar de ejecución de Java y proporciona un mecanismo simple para la comunicación de servidores en aplicaciones distribuidas basadas exclusivamente en Java.

SOAP (Simple Object Access Protocol): es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML.

SSH (Secure SHell): es un protocolo que facilita las comunicaciones seguras entre dos sistemas usando una arquitectura cliente/servidor y que permite a los usuarios conectarse a un host remotamente.

Triggers: también conocido como desencadenador, éste se ejecuta des atendidamente y automáticamente cuando un usuario realiza una acción con la tabla de una base de datos que lleve asociado esta operación (Operaciones DML).

UDDL (Universal Description, Discovery and Integration): es un registro público diseñado para almacenar de forma estructurada información sobre empresas y los servicios que éstas ofrecen.

WSDL (Web Services Description Language): es un formato XML que se utiliza para describir servicios Web.