Universidad de las Ciencias Informáticas

Facultad 6

Título: “BioG: Desarrollo de un módulo de simulación de sistemas dinámicos”

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

Autor: Katia Manchón Solano

Tutor: MSc. Noel Moreno Lemus

Ing. Niurka Martínez Durán

Ciudad de la Habana
Junio - 2009

"El futuro de Cuba debe ser necesariamente un futuro de hombres de ciencia..."

Fidel Castro Ruz.

DECLARACION DE AUTORIA

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

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

Katia Manchón Solano	**MSc. Noel Moreno Lemus**	**Ing. Niurka Martínez Durán**
_____	_____	_____
Firma del Autor	Firma del Tutor	Firma del Tutor

DATOS DE CONTACTO

Tutor:
MSc. Noel Moreno Lemus
1

Tutor:
Ing. Niurka Martínez Duran
nduran@uci.cu

DEDICATORIA

Con todo el amor del mundo dedico esta tesis a:

Mis padres Andrea Solano y Francisco Manchón, por haberme guiado siempre por el camino correcto de la vida, por estar siempre a mi lado, educarme sin descanso, por todo su apoyo, por sus consejos oportunos y su entrega sin límites; especialmente a mi mamita quien ha sido mi guía incondicional, por su amor, dedicación y preocupación constante.

A Dayan por haber estado siempre conmigo en mis últimos años de la carrera, por su incansable preocupación, por apoyarme y ayudarme en todo momento.

A mis hermanos, en especial a Arael por su atención constante.

AGRADECIMIENTOS

A mi familia por brindarme todo su apoyo.

A mis tutores: MSc. Noel Moreno Lemus e Ing. Niurka Martínez Durán por su atención y sus asesorías constantes, por estar siempre presentes cuando los necesitaba.

A Veylys Capote Serrano, por su ayuda incondicional en la creación de esta tesis.

A Keiler, mi Jefe de Proyecto por su gran ayuda y asesorías constantes.

A Maritza Pérez y Rigoberto Hernández, por estar siempre presente y apoyarme en todo momento.

Agradezco infinitamente a todas las personas que de una forma u otra me ayudaron y me apoyaron a lo largo de mi carrera.

A la Revolución y a Fidel, por haber creado este proyecto futuro del cual hoy formo parte.

Gracias a todos.

RESUMEN

El software para la Simulación de Sistemas Biológicos (alas BioSyS) es una aplicación de propósito general producto del desarrollo del software independiente que permite modelar, editar ecuaciones, estimar parámetros, simular tanto de forma local como distribuida y hacer Meta-análisis de los resultados obtenidos en las simulaciones a modelos matemáticos asociados a sistemas biológicos que puedan ser descritos mediante sistemas de ecuaciones diferenciales.

Este trabajo centra su atención en la realización de una Aplicación Web para gestionar las funcionalidades básicas del software alas BioSyS. El sistema fue llevado a cabo siguiendo los pasos que propone el Proceso Unificado de Desarrollo de Software y empleándose para el modelado del mismo la herramienta CASE Visual Paradigm; se utilizaron tecnologías de software libre: java como plataforma y lenguaje de programación.

PALABRAS CLAVE

- alas BioSyS
- Aplicación Web

TABLA DE CONTENIDOS

INTRODUCCION

Los Sistemas Dinámicos surgen a partir de la necesidad de entender cómo evolucionan los procesos de la naturaleza. El estudio de estos sistemas va, desde los más diversos enfoques de la física y las matemáticas, hasta sus posibles aplicaciones a las diferentes áreas de actividad humana. Un sistema dinámico es un sistema complejo que presenta un cambio o evolución de su estado en el tiempo, el comportamiento en dicho estado se puede caracterizar determinando los límites del sistema, los elementos y sus relaciones; de esta forma se puede elaborar modelos que buscan representar la estructura del mismo. [1]

Este enfoque ha sido ampliamente utilizado en la Biología de Sistemas, que no es más que una rama de las ciencias biológicas que intenta describir los mecanismos que regulan la vida de los seres vivos de forma que, conocida la descripción, se pueda predecir qué va a ocurrir en una célula, tejido o ser vivo. [2] Lo que se pretende es en el futuro poder describir a los seres vivos mediante modelos matemáticos que puedan predecir el funcionamiento de los mismos.

La Biología de Sistemas integra conceptos de diferentes ramas de la ciencia como por ejemplo la Bioquímica, la Genómica, las Matemáticas, la Biología Molecular y las Ciencias Ingenieriles. Además permite estudiar los sistemas biológicos a diferentes niveles de abstracción (células, tejidos, ecosistemas, etc.) y haciendo uso de diferentes herramientas matemáticas (ecuaciones probabilísticas, ecuaciones en diferencias, autómatas celulares, sistemas de ecuaciones diferenciales, etc.). Los sistemas de ecuaciones diferenciales, por las facilidades que brindan tanto desde el punto de vista de la modelación, como por la cantidad de algoritmos y herramientas computacionales desarrolladas para el trabajo con los mismos, son los más utilizados.

El enfoque sistémico ha ido invadiendo la actividad de los muchos centros de investigación del mundo que se dedican al estudio de los sistemas biológicos. Cuba no se ha quedado fuera de Revolución e instituciones como el Centro de Ingeniería Genética y Biotecnología (CIGB) y el Centro de Inmunología Molecular (CIM) utilizan este enfoque en sus investigaciones.

Pero desafortunadamente estas instituciones no cuentan con herramientas computacionales para la modelación, comparación y simulación de Sistemas Biológicos. El elevado precio de las mejores aplicaciones de software desarrolladas en esta área, los requerimientos de hardware de las mismas, y la no adaptación a las necesidades de los modelos matemáticos que en estos centros se desarrollan, constituyen algunos de los motivos por los cuales se adolece de este tipo de herramienta. La Universidad

de Ciencias Informáticas conjuntamente con el Centro de Inmunología Molecular (CIM), se encuentran desarrollando un software que permitirá la integración de todas las funcionalidades básicas para la biosimulación, con el fin de facilitar la obtención y producción de nuevos biofármacos: Biological System Simulator (alas BioSyS).

La aplicación desktop alas BioSyS está formada por diferentes módulos que permiten:

- Modelar gráficamente sistemas biológicos
- Crear modelos matemáticos formados por Sistemas de Ecuaciones Diferenciales.
- Realizar Simulaciones de los modelos matemáticos previamente creados.
- Realizar estimaciones de parámetros a partir de un conjunto de datos experimentales.
- Realizar determinados tipos de análisis como:
- Análisis de clústeres
- Clasificaciones

A pesar de existir una versión 1.0 de este producto y de contar con todas las funcionalidades antes mencionadas alas BioSyS como aplicación desktop presenta limitaciones al requerir de grandes recursos computacionales para su funcionamiento, necesitar servidores de Base de Datos y ser extremadamente complicado de instalar y configurar, por lo que solo un grupo limitado de los especialistas de la Biología de Sistemas podrán contar con el mismo funcionando al 100 % de sus potencialidades.

A partir de un análisis de cómo utilizar la tecnología de la información para organizar, y distribuir esta aplicación surge el siguiente **problema científico**: ¿Cómo lograr que las funcionalidades básicas de alas BioSyS estén accesible a la comunidad interesada en la Biología de Sistemas? Para ello el problema se enmarca en el **objeto de estudio**: Las Aplicaciones Web. El **campo de acción** que comprende el mismo es: Las Aplicaciones Web para Sistemas Dinámicos. El **objetivo general** para dar solución a la problemática planteada es: Desarrollar una aplicación Web, para gestionar las funcionalidades básicas de alas BioSyS. De los cuales se derivan los siguientes **objetivos específicos**:

- Definir las funcionalidades que tendrá la aplicación alas BioSyS Web.
- Diseñar la arquitectura candidata de la aplicación Web para gestionar las funcionalidades de alas BioSyS seleccionadas.
- Implementar la aplicación Web para gestionar las funcionalidades de alas BioSyS seleccionadas.

Tareas a desarrollar:

1. Investigación del estado del arte para las aplicaciones Web de simulación de sistemas biológicos.
2. Investigación de las tendencias y tecnologías en las aplicaciones Web de simulación de Sistemas Biológicos.
3. Análisis y selección de las tecnologías para desarrollar la aplicación Web.
4. Realización de las actividades del flujo de trabajo de análisis y diseño.
5. Realización de las actividades del flujo de trabajo de implementación.

Con el desarrollo de esta aplicación Web se les facilitará a los usuarios acceder a un software de Sistemas Biológicos y obtener los resultados correspondientes.

CAPITULO 1: FUNDAMENTACION TEORICA

El presente capítulo brinda una descripción general de la Biología de Sistemas. Se hace referencia a los conceptos Biología de Sistemas, Aplicaciones Web, Portlets. Describe las características de las diferentes aplicaciones web desarrolladas en el mundo para el estudio de los Sistemas Biológicos. Se establece la metodología de trabajo, las herramientas y técnicas que se usarán en la investigación.

1.1 Aplicaciones Web

El concepto de aplicación Web difiere de cada desarrollador, pero de manera general se podría decir que, es un sistema informático que permite al usuario acceder a través de una interfaz interactiva a los recursos y servicios ofrecidos por un servidor web, puede utilizarse como enlace una intranet o Internet. Esta comunicación no se trata solamente de páginas web estáticas, sino que incluye páginas web dinámicas, por tanto, requieren de tecnologías que proporcionen la lógica de la generación de contenido dinámico del lado del servidor (php, asp, jsp). Utilizan además lenguajes interpretados del lado del cliente (HTML, XML, Ajax) y se combinan para aprovechar su potencial y posibilitar una solución más extensible y portable. [3]

En las aplicaciones Web el software es almacenado en el servidor y todas o algunas partes del mismo se descargan desde la Web cada vez que se ejecuta por medio de navegadores Web. Estas aplicaciones facilitan a los usuarios, migrar de sistema operativo o cambiar libremente el hardware sin que el funcionamiento de ellas sea afectado. Al solo necesitar un navegador implica que no hay que distribuirlas entre los demás computadores de modo que desde cualquier lugar con conexión de red se puede trabajar.

Las aplicaciones Web contienen un conjunto de páginas Web estáticas y dinámicas. Una página Web estática es aquélla que no cambia cuando un usuario la solicita, el servidor Web envía la página al navegador Web solicitante sin modificarla. Por el contrario, el servidor modifica las páginas Web dinámicas antes de enviarlas al navegador solicitante. La naturaleza cambiante de este tipo de página es la que le da el nombre de dinámica.

Existen muchas variaciones posibles para la estructura de una aplicación web, pero normalmente está estructurada como una aplicación de tres-capas. En su forma más común, el navegador web ofrece la primera capa y un motor capaz de usar alguna tecnología web dinámica para generar las páginas web

dinámicas, constituye la capa intermedia.

Las páginas web dinámicas son modificadas por el servidor antes de enviarlas al navegador solicitante, que a diferencia de las estáticas no cambian cuando un usuario las solicita. La tercera y última capa la compone, una base de datos. El navegador web manda peticiones a la capa intermedia que ofrece servicios valiéndose de consultas y actualizaciones a la base de datos, a su vez proporciona una interfaz de usuario.

Entre los diversos tipos de aplicación Web se encuentran las informacionales, interactivas, comunitarias, transaccionales, workflow, portales y portlets. Algunos ejemplos de aplicaciones Web son:

- Escritorio virtual.
- Correo electrónico.

1.2 Portlet

1.2.1 Definición de Portlet

Un Portlet es un componente Web manejado a través de un contenedor de Portlets que procesa las peticiones de los clientes y produce contenido dinámico. [4] El contenido generado por un Portlet es llamado fragmento, una pieza de código HTML, XHTML, WML adherida a ciertas reglas. Un fragmento puede ser agregado a otros fragmentos a fin de formar un documento completo, por ejemplo, un portal es un conjunto de fragmentos generados por diversos Portlets.

El contenido generado por un Portlet puede variar de un usuario a otro dependiendo de cómo haya configurado el usuario el Portlet. Los Portlets no tienen interacción directa con los clientes Web, en su lugar los clientes Web interactúan con el portal a través de un mecanismo de solicitud/entrega aplicado por un contenedor de Portlets el cual también maneja el ciclo de vida de los Portlets.

Generalmente, los Portlets tienen una clara separación entre el contenido y la presentación, la cual es manejada por una o más clases de Java que contienen la aplicación lógica. Los portales usan los Portlets como componentes modulares para interfaz de usuario. Surgen como respuesta a la necesidad de simplificar el uso de dichas aplicaciones haciendo uso de interfaces amigables.

1.2.2 Funcionamiento del Portlet

Teniendo en cuenta ciertas condiciones de proceso, un Portlet procesa una petición y responde con la salida apropiada a cada caso. Si se trata de un portal con varios módulos, la encargada de gestionar y controlar la ejecución, comunicación y respuesta de todos los Portlets activos, es la interfaz de programación de aplicaciones API (Applications Programming Interface).

Naturalmente, también es posible procesar peticiones mediante un protocolo simple de acceso a objetos SOAP (Simple Object Access Protocol) dentro de un Portlet a través de aplicaciones de terceros.

Los Portlets desempeñan diferentes tareas y crean contenidos de acuerdo a su función actual. Un modo de Portlet indica la función que está desempeñando en cierto momento, especifica el tipo de tarea que debería desempeñar y qué contenido debería generar. Cuando se invoca a un Portlet, el contenedor de Portlets provee el modo para el actual requerimiento al Portlet. Estos pueden cambiar su modo mientras procesan una petición de acción.

1.3 Biología de Sistemas

"La Biología de Sistemas es un área de las ciencias biológicas que integra conceptos e ideas de las ciencias de la vida, las ciencias de la computación y disciplinas ingenieriles." [5] Dicha área ha ido transformando nuestra manera de pensar sobre los sistemas biológicos, teniendo siempre el enfoque sistémico por delante, algunas veces cerca de la genómica, otras explorando áreas desconocidas o poco estudiadas. [6] Este cambio de paradigma tiene una implicación inmediata para las investigaciones médicas y la prevención de enfermedades que hasta ahora se han resistido a diferentes tratamientos.

Resumiendo, podemos decir que el enfoque sistémico será capaz de revolucionar, no solo nuestro conocimiento en las ciencias biológicas básicas, sino que tiene una implicación directa en los resultados futuros de las investigaciones en ciencias aplicadas como la medicina y la biotecnología.

Las investigaciones en Biología de Sistemas se pueden dividir en dos grandes áreas interrelacionadas, una experimental y otra computacional. Aunque en ambas áreas ha habido un desarrollo notable, ha sido el lado computacional el que más ha evolucionado. Así ya vemos cómo la necesidad de

desarrollar herramientas informáticas en este campo, ha ido llegando a los sectores comerciales de la biotecnología.

Recientemente en los Estados Unidos se han creado las primeras empresas de biología de sistemas cuyos productos estén intrínsecamente relacionados con la simulación de problemas biológicos. Ejemplo de tales empresas son:

- Gene Network Sciences (www.gnsbiotech.com, fundada 2000).
- Entelos (www.entelos.com, fundada 1996), Physiome (www.physiome.com, fundada 2001).
- Genomática (www.genomatica.com, fundada 2001).

El desarrollo de software de forma independiente también ha ido en ascenso. Así en la página Web del SBML http://www.sbml.org3 podemos encontrar referencias a más de 100 software basados en este estándar que permiten modelar, simular o realizar análisis sobre modelos de sistemas biológicos.

Todos estos software trabajan con algún tipo de modelo, ya sea gráfico o matemático, que les permite representar de una forma relativamente simple el problema en estudio. Estas herramientas informáticas han sido de gran ayuda para los científicos que se dedican a estudiar sistemas biológicos. No obstante, existen ciertas desventajas que bien vale la pena discutir. Por ejemplo, las bases de datos que guardan información de los sistemas biológicos se centran, por lo general, en almacenar todo lo relacionado con los modelos, ya sean matemáticos o gráficos, pero no almacenan información de los estudios que sobre estos se realizan. Esto implica que, aunque podamos disponer de grandes repositorios de modelos, si queremos saber cómo funcionan dichos sistemas, tenemos que repetir el trabajo que ya otros han realizado. [7]

Otro ejemplo concreto es el caso de las herramientas de simulación y análisis. Los desarrolladores de las primeras se han centrado en crear sistemas muy robustos en cuanto a la modelación, pero, que poco hacen a la hora de simular, pues muchos de estos software solo permiten realizar estudios elementales de dinámicas de poblaciones, bifurcaciones obligando al usuario a estar sentado durante horas frente a la maquina, interactuando con el sistema e interpretando la infinidad de resultados que de él obtiene.

Solo algunas aplicaciones ya implementan algoritmo para la realización de exploraciones intensivas sobre los modelos matemáticos de los sistemas en estudio, permitiendo al usuario centrarse en la

interpretación de los resultados finales y eliminando el tedioso trabajo artesanal. Este tipo de sistemas utilizan por lo general infraestructuras distribuidas para realizar el gran volumen de cálculos que se requieren.

Estas desventajas limitan en gran medida el desarrollo de las investigaciones en el área de la Biología de Sistemas, pues los especialistas dedican gran parte de su tiempo a estar sentados frente a estos software, haciendo trabajos que los mismos podrían hacer perfectamente de forma automática. [7]

A partir de dichas desventajas, surge el software alas BioSyS, permitiendo eliminar dichas limitantes debido a que la aplicación permite realizar exploraciones intensivas en sistemas biológicos descritos mediante sistemas de ecuaciones diferenciales. Almacena toda la información generada en una base de datos y realiza un análisis de los resultados obtenidos, además de incorporar una serie de funcionalidades básicas como: análisis de bifurcaciones, cálculo de estados de equilibrio, etc.

Lo más interesante de BioSyS es la implementación de una serie de Meta-análisis, que haciendo uso de técnicas de Inteligencia Artificial y minería de datos brinda la posibilidad de extraer de su base de datos informaciones muy valiosas sobre el comportamiento de los sistemas en estudio.

A pesar de todas estas ventajas existen algunas desventajas que dificultan su utilización:

La primera de ellas es la dependencia de otros software: los asistentes matemáticos Matlab y Máxima, los gestores de base de datos MySQL y PostgreSQL y el software para minería de datos, weka. También hace uso de la plataforma de tareas distribuidas t-arenal para la resolución de aquellos problemas que son costosos computacionalmente.

La segunda es que, para explotar todas sus potencialidades se requiere de grandes recursos de cómputo, a los que el usuario interesado no tiene por qué tener acceso. En resumen, su instalación y configuración es un proceso complejo de llevar a cabo y solo podrá ser ejecutado por especialistas capacitados.

BioSyS no es el único caso que presenta esta limitante, es por ello que una tendencia ha sido el desarrollo de aplicaciones Web, pues en este caso los que despliegan estas aplicaciones son los que tienen que contar con estas complejas infraestructuras, no así el usuario final, el cual solo necesita de un browser y una conexión a internet, como ya se mencionó anteriormente.

A continuación se describen algunas de las aplicaciones web desarrolladas para Biología de Sistemas.

1.4 Aplicaciones de Software para la Simulación de Sistemas Biológicos

En la actualidad, a nivel internacional se han desarrollado varias aplicaciones de software para el estudio de Sistemas Biológicos:

ImmunoGrid

ImmunoGrid: es una implementación virtual del sistema inmune humano que simula procesos inmunes. Es capaz de simular procesos a escala natural y proporciona herramientas para el diseño de vacunas. Lo novedoso está en lograr la conexión de las interacciones a nivel molecular con los modelos a nivel de sistema. Hace uso de bases de datos para almacenar, manipular y modelar datos inmunológicos, lo que facilita la búsqueda de modelos matemáticos y predicativos.

Limitaciones: Se centra sólo en problemas y enfermedades del sistema inmune. Los tipos de modelos matemáticos son complejos, hay que definirlos solamente en formato de texto plano. La aplicación Web que se diseñará no sólo debe centrarse en problemas y enfermedades del sistema inmune, sino que debe incluir cualquier problema biológico que pueda ser modelado mediante sistemas de ecuaciones diferenciales.

Cellware

Cellware no sólo se ha diseñado para conducir la modelación y simulación de los caminos reguladores y metabólicos del gen, sino también para ofrecer un ambiente integrado para la diversidad de representaciones matemáticas, la valoración del parámetro y la optimización. Además, tiene una exhibición gráfica y una capacidad de uso fácil para trabajar con modelos grandes y complejos que serían proporcionadas por defecto.

Limitaciones*:* No hace análisis exhaustivo de las simulaciones realizadas, es decir, aunque permite realizar múltiples análisis, la interpretación de los resultados queda en manos de los usuarios. La aplicación no sólo debe conducir la modelación y simulación sino que además permitirá realizar análisis profundos de los Sistemas Biológicos que se simulan, facilitando el proceso de interpretación de los resultados por parte de los investigadores.

Webcell

Webcell: es un ambiente de la simulación integrado para manejar la información cuantitativa y cualitativa sobre las redes celulares, para explorar su estado-sostenido y las conductas dinámicas interactivamente sobre la red.

Una interfaz amigable, les permite eficazmente a los usuarios crear, visualizar, simular y guardar sus modelos de red de reacción, mientras se facilita la modelación cinética y la simulación de Sistemas Biológicos de interés. Los métodos de análisis apoyados para tales modelos incluyen, el análisis de la senda estructural, el análisis del mando metabólico (MCA), análisis de conservación y simulación dinámica.

Limitaciones*:* Se centra solo en la dinámica del sistema, no permitiendo realizar otro tipo de análisis. La aplicación que se diseñará no sólo mostrará la dinámica del sistema sino que permitirá al usuario realizar innumerables simulaciones a través de una interfaz amigable. Permitirá a través de algoritmos ya implementados, el análisis de todas las simulaciones realizadas.

Vcell

Vcell permite crear modelos biológicos del cual se generará un código matemático que podrá ser representado gráficamente en 1D, 2D, 3D y que es necesario para correr las simulaciones que pueden ser complejas, variando uno o muchos parámetros según una lista especificada de valores posibles o de un rango definido.

Limitaciones: No realiza análisis de los resultados.

Aunque estas aplicaciones resuelven en alguna medida el problema de las infraestructuras de almacenamiento y cómputo, siguen teniendo desventajas.

Por ejemplo, debido a la cantidad de información que se maneja en el estudio de los Sistemas Biológicos, el desarrollo de un sistema que gestione información para la simulación de Sistemas Biológicos, requiere de varias interfaces que se comuniquen entre sí. Desarrollar una aplicación Web que conste de dichas interfaces, sería una de las vías para resolver el problema. La desventaja de esta solución es que la persona que interactúe con el sistema podría realizar solamente la gestión de un tipo de información, para luego gestionar otra.

El desarrollo de Portlets contribuye a eliminar esta limitante, debido a que estos brindan la oportunidad, mediante una sola interfaz, de poder gestionar varias informaciones a la vez. Desde el punto de vista del usuario final, los Portlets ayudan de una mejor manera la usabilidad de un portal, ya que cada usuario podrá ajustar a su manera la distribución y la apariencia de los componentes con los que desea trabajar.

El administrador del portal podrá decidir qué puede ver y qué no, cada usuario. Esta característica puede lograrse, evidentemente, sin la necesidad del uso de Portlets, pero implica para el desarrollador mayor tiempo de trabajo, que puede simplificarse con la utilización de Portlets, sin tener en cuenta que, además, se estará siguiendo un estándar abierto. Por esta razón se propone desarrollar un Portlet que permita la gestión de la información necesaria para la simulación de Sistemas Biológicos sirviéndonos del portal GridSphere de Servicios Bioinformáticos, que ha sido implementado con anterioridad. [3]

1.5 Metodologías, Tecnologías y Herramientas utilizadas

Todo desarrollo de software es arriesgado y difícil de controlar, si además no media una metodología definida, conlleva al resultado de clientes y desarrolladores insatisfechos. Es necesario definir, qué tecnologías y herramientas se deben utilizar para el desarrollo de una aplicación de esta envergadura. Se presenta una explicación acerca de estos puntos, los cuales describen la conformación de esta aplicación.

1.5.1 Metodologías de Desarrollo de Software

Los desarrolladores se ven comprometidos en la realización de un software al enfrentarse con la necesidad de desarrollar con calidad y en el menor tiempo posible. Siendo inmediato el conocimiento referente a la organización de las actividades para cada desarrollador por separado y para el equipo de desarrollo. Por tanto, se requiere de una metodología para dirigir las actividades vinculadas al proceso de desarrollo de software a fin de añadir robustez a la producción.

Por una parte tenemos aquellas propuestas más tradicionales que se centran especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir y las herramientas y notaciones que se usarán. Estas propuestas han demostrado ser efectivas y necesarias en un gran número de proyectos, pero también han presentado problemas en

otros muchos.

Una posible mejora es incluir en los procesos de desarrollo más actividades, más artefactos y más restricciones, basándose en los puntos débiles detectados. Sin embargo, el resultado final sería un proceso de desarrollo más complejo que puede incluso limitar la propia habilidad del equipo para llevar a cabo el proyecto.

Otra aproximación, es centrarse en otras dimensiones, por ejemplo, el factor humano o el producto software. Esta es la filosofía de las metodologías ágiles, las cuales dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas. Este enfoque está mostrando su efectividad en proyectos con requisitos muy cambiantes y cuando se exige reducir drásticamente los tiempos de desarrollo, pero manteniendo una alta calidad.

De tantas metodologías con tantos enfoques para dirigir las actividades vinculadas al proceso de desarrollo de software, se estudiaron las siguientes:

XP (Extreme Programing o Programación extrema)
Lo fundamental en este tipo de metodología es:

- La comunicación, entre los usuarios y los desarrolladores.
- La simplicidad, al desarrollar y codificar los módulos del sistema.

La retroalimentación, concreta y frecuente del equipo de desarrollo, el cliente y los usuarios finales.

Es una de las metodologías de desarrollo de software más exitosas en la actualidad utilizada para proyectos de corto plazo, corto equipo y cuyo plazo de entrega es sumamente mínimo, esto es una desventaja concreta para desarrollar la presente aplicación, ya que el mismo es, conforme a sus consideraciones, un proyecto extenso y requiere un plazo de tiempo mayor que el estipulado en esta metodología presentada.

La filosofía de XP es satisfacer al completo las necesidades del cliente, por eso lo integra como una parte más del equipo de desarrollo, haciéndolo inaplicable para el desarrollo de la presente aplicación, ya que el cliente no se puede ver de otra forma que como agente externo al equipo de trabajo. [8]

Microsoft Solution Framework (MSF)

MSF: es una metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, los cuales representan un compendio de las mejores prácticas en cuanto a administración de proyectos se refiere. Y tiene las siguientes características:

- Adaptable: es parecido a un compás, usado en cualquier parte como un mapa del cual su uso es limitado a un lugar específico.
- Escalable: puede organizar equipos tan pequeños entre 3 ó 4 personas, así como proyectos que requieren 50 personas o más.
- Flexible: es utilizada en el ambiente de desarrollo de cualquier cliente.
- Tecnología Agnóstica: porque puede ser usada para desarrollar soluciones basadas sobre cualquier tecnología. [9]

Una de las desventajas que se constató en el estudio realizado acerca de MSF, es que en cada fase se debe documentar cada cosa que se haga, esto indudablemente atenta al equipo de producción que quiera agilizar el desarrollo del software. Otra deficiencia que se pudo apreciar es que con frecuencia no ofrece ninguna orientación sobre cómo llegar a un objetivo determinado, tales como: producir un artefacto, realizar alguna tarea específica, entre otras; lo cual implica un riesgo que no le agradaría encontrar a ningún equipo de desarrollo durante la realización de una aplicación de software.

RUP (Proceso unificado de desarrollo de software)

Las cualidades que denotan más méritos en RUP son las siguientes:

- Dirigido por los casos de uso: Parte de la idea que un sistema debe brindar todos los servicios que requiere el usuario. Define caso de uso como el conjunto de acciones que debe realizar el sistema para devolverle al usuario un resultado de valor. Siendo esto la directriz que guía los pasos del desarrollo del software.
- Centrado en la arquitectura: Se toma el diseño completo ignorando los detalles, centrándose en las características más importantes. Se evalúa no solo las necesidades de los usuarios o inversores, sino también todos los aspectos técnicos que rodean y permiten que el sistema

funcione con toda eficiencia. Así la arquitectura va evolucionando en su interacción con los casos de uso hasta llegar a un equilibrio entre funcionalidad y características técnicas.

- Iterativo e incremental: La alta complejidad de los sistemas actuales hace que sea factible dividir el proceso de desarrollo en varios mini-proyectos. Cada uno de éstos se les denomina iteración y pueden o no representar un incremento en el grado de terminación del producto completo. En cada iteración los desarrolladores seleccionan un grupo de casos de uso, los cuales se diseñan, implementan y prueban. La planificación de iteraciones hace que se reduzcan los riesgos de los costos de un solo incremento, no sacar al mercado un producto en el tiempo previsto, mantener la motivación del equipo, pues puede ver avances claros a corto plazo y que el desarrollo pueda adaptarse a los cambios en los requisitos.

Estas características van a permitir una planificación ajustada al avance que vaya presentando el producto. Además, se irá verificando que las funcionalidades que van siendo implementadas en el software sean exactamente las que desea el usuario. El Proceso Unificado del Rational aporta además un enfoque disciplinado a la asignación y dirección de tareas y responsabilidades, tanto individuales como al equipo de trabajo. Una característica importante es que permite corregir errores en cada iteración y es flexible a cambios en los requerimientos. [10]

Open UP

Open UP conserva las características principales del modelo de desarrollo RUP como una extensión del mismo. Incluye el desarrollo iterativo. Permite identificar los requisitos operacionales del sistema. Prevé las interacciones con los usuarios y los posibles riesgos en el desarrollo del sistema. Plantea que se debe tener una versión ejecutable del proyecto en poco tiempo.

En esta metodología solo se debe usar los procesos que sean necesarios y también no se debe usar muchos artefactos, además debe acoplarse a las necesidades del usuario pudiendo ser modificado y extendido. Usa las mismas fases e hitos del RUP pero sin tanta documentación. Se enfoca en el valor del software y disminuye riesgos.

Open UP es una metodología de desarrollo ágil y ligera, consiste en equipos a los cuales se les asigna una fase del desarrollo que tienen que complementarse entre sí para obtener un buen producto final, no puede ser una sola persona la que realice todo el trabajo, pues esto podría ocasionar que se pierda de vista ciertas características importantes, por ejemplo para un proyecto pequeño, Open Up

constituye equipos de 3 a 6 personas e implica de 3 a 6 meses de esfuerzo del desarrollo. [11]

Por las razones descritas anteriormente, además de ser la metodología escogida por el polo de la facultad, se eligió Open Up como la metodología de desarrollo para la continuidad de la aplicación.

1.5.2. Herramientas Case

(CASE: Computer Aided Software Engineering o Ingeniería de Software Asistida por Computadoras)

Estas herramientas están elaboradas para aumentar la productividad en el desarrollo de software. Además, reduce el coste del mismo en términos de tiempo y dinero. Pueden guiar en todos los aspectos del ciclo de vida de desarrollo del software, ya sea en el diseño, cálculo de costos, generación de código automático según un diseño dado, compilación automática, documentación o detección de errores, entre otras.

Para la selección correcta de la herramienta CASE a aplicar, se ha procedido a un estudio de aquellas que han demostrado mayor eficiencia y que a la vez sean ampliamente usadas. Encontrando que las más utilizadas y eficientes apuntan a ser Racional Rose y Visual Paradigm.

La primera herramienta citada mantiene la consistencia de los modelos del sistema de software, genera automáticamente la documentación y el código a partir de modelos; pero se descartó, ya que se vio una carencia de integración cuando se incorpora algunas funcionalidades a través de otras aplicaciones. Además, el equipo de desarrollo, al implementar directamente sobre la plataforma GNU/LINUX (sistema operativo Ubuntu) no cuenta con la versión de Racional Rose que corra sobre dicha plataforma.

Visual Paradigm: es una herramienta CASE potentísima para visualizar y diseñar elementos de software, para ello utiliza el lenguaje UML y proporciona a los desarrolladores una plataforma que les permita diseñar un producto con calidad de una forma rápida. [12] Está disponible en varias ediciones:

Enterprise, Professional, Community, Standard, Modeler y Personal. Genera código y realiza ingeniería inversa para varios lenguajes de programación: Java, C++, CORBA IDL, PHP, XML Schema y ADA.

Facilita la interoperabilidad con otras herramientas CASE como el Rational Rose y se integra con las siguientes herramientas Java: Eclipse/IBM WebSphere, Jbuilder, NetBeans IDE, Oracle Jdeveloper,

BEA Weblogic. Además genera código para C#, Visual Basic.net, Object Definition Lenguage (ODL), Flasch Action Script, Delphi, Perl y Phyton. Se integra con el Visio para importar imágenes del mismo y realizar los diagramas de despliegue. Genera documentación para el proyecto en HTML, MS Word y PDF. Además exporta e importa los diagramas en el estándar XML y como imágenes (ya sea con extensiones .jpg o .png). Es gratis en su edición Community y multiplataforma.

Convenientemente al sistema operativo sobre el cual el equipo de desarrollo trabaja (distribución de GNU/LINUX: Ubuntu) se decidió utilizar Visual Paradigm como herramienta CASE para visualizar y diseñar los elementos de software, ya que es multiplataforma y utiliza el Lenguaje Unificado de Modelado (UML). Además, el desempeño sobre su uso es mucho más sencillo y ágil que el del Rational Rose.

1.5.3. Lenguaje de Modelado del Software

UML (Unified Modeling Languaje o Lenguaje Unificado de Modelado)

Surgió en 1994 como la unificación de todos los métodos de diseño anteriores. Con él se visualiza, especifica y documenta cada una de las partes que comprende el desarrollo de software.

Es posible fijar la serie de requerimientos y estructuras necesarias para determinar un sistema de software previo al proceso de implementación del código. Se presta por su naturaleza para el intercambio entre los usuarios y desarrolladores. Ha llegado a convertirse, casi plenamente, en el lenguaje estándar del análisis y diseño de sistemas informáticos dentro de la industria del software. [13]

Anteriormente los revisores de algún sistema debían enfrentarse a aprender las semánticas y notaciones de la metodología empleada antes de entender el diseño en sí. UML permite que diseñadores diferentes modelen sistemas diferentes y puedan ampliamente entender cada uno los diseños de los otros.

1.5.4. Lenguaje de Programación

Java, utiliza el concepto de máquina virtual (VM). El código que se genera no es específico a una plataforma en particular. Un programa nativo: la VM se encarga de traducir este código para que

cualquier ordenador pueda ejecutarlo. De esta manera un código generado en Java puede correr en cualquier plataforma, donde se haya portado la VM. Se utilizará Java como lenguaje de programación para desarrollar la herramienta por ser un lenguaje cuya portabilidad está verdaderamente probada y no requerir largos periodos de tiempo para el desarrollo de las aplicaciones. Otras características de dicho lenguaje son:

- Orientado a Objetos: Java trabaja con sus datos como objetos y con interfaces a esos objetos, soporta las características propias del paradigma orientado a objetos: abstracción, encapsulamiento, herencia y polimorfismo.
- Simple: Posee una curva de aprendizaje muy rápida. Ofrece toda la funcionalidad de un lenguaje potente, pero sin las características menos usadas y más confusas de éstos.
- Robusto: Java realiza verificaciones en busca de problemas, tanto en tiempo de compilación como en tiempo de ejecución. La comprobación de tipos en Java ayuda a detectar errores lo antes posible en el ciclo de desarrollo. Java obliga a la declaración explícita de los tipos de los ítems de información, reduciendo así, las posibilidades de error. Maneja la memoria para eliminar las preocupaciones por parte del programador de la liberación o corrupción de la misma. [14]

1.5.5. Sistema de Control de Versiones

Una versión de un producto: es el estado en que se encuentra en un momento dado de su desarrollo o modificación. Se llama control de versiones a la gestión de los diversos cambios o configuraciones que se realizan sobre los elementos de algún producto. [15]

Los sistemas de control de versiones facilitan la administración de las distintas versiones de cada producto desarrollado, controlando el acceso a ficheros que están bajo su cuidado. El código fuente de un programa necesita controlarse en cuanto a su almacenaje, posibilidad de realizar cambios, registro histórico de las acciones realizadas y, tal vez, la generación de informes que traten sobre el estado y los cambios introducidos entre las dos versiones.

Todo es posible hacerse a través del uso de un sistema de control de versiones. Opera generalmente con una copia maestra en un repositorio central y un programa cliente con el que cada usuario

sincroniza su copia local. Permitiendo compartir los cambios sobre un mismo conjunto de ficheros y guardar el registro de los cambios realizados por cada usuario, admitiendo así el retorno seguro al estado anterior en caso de necesidad.

Constituye una garantía de seguridad y una gran utilidad, contar en el proyecto con un sistema de control de versiones. Ilustrando en la práctica, se verá muy frecuente la necesidad de un sistema control de versiones. Cuando se está escribiendo un programa o un documento muy largo (como un informe o una tesis) y se quiere hacer una marca en el proyecto al llegar a un punto estable, éste permite guardar una versión estable del trabajo.

La idea de guardar versiones es, que si los nuevos cambios están mal, la marca servirá para volver al punto estable y recomenzar desde ahí. Si los cambios están bien y se alcanza una nueva estabilidad, se sitúa otra marca. Para el control de versiones de esta herramienta se utilizará Subversion (SVN). El Concurrent Versions System (CVS) es el padre de los controladores de versiones, pero el mismo fue reemplazado por SVN debido a que el primero posee varias deficiencias. Por tanto el SVN es el indicado para un desarrollo con alta competitividad y calidad, destacando en el mismo:

- Es software libre bajo una licencia de tipo Apache/BSD.
- A diferencia de CVS, los archivos versionados no tienen cada uno un número de revisión independiente. En cambio, todo el repositorio tiene un único número de versión que identifica un estado común de todos los archivos del repositorio en cierto punto del tiempo.
- Permite selectivamente el bloqueo de archivos.
- Maneja eficientemente archivos binarios (a diferencia de CVS que los trata internamente como si fueran de texto).
- Se envían sólo las diferencias en ambas direcciones (en CVS siempre se envían al servidor archivos completos).
- La creación de ramas y etiquetas es una operación más eficiente. Tiene costo de complejidad constante (O (1)) y no lineal (O(n)) como en CVS.
- Las modificaciones (incluyendo cambios a varios archivos) son atómicas.

- Se sigue la historia de los archivos y directorios a través de copias y renombrados. [15]

1.5.6. Entorno de Desarrollo

Para el lenguaje de programación Java existen varios entornos de desarrollo integrado (IDE del inglés Integrated Development Environment) que permiten su uso, tales como: JBuilder, NetBeans y Eclipse. Una de las deficiencias más críticas que desclasifican a JBuilder como IDE para el presente software es que no es multiplataforma, solo corre sobre Windows.

NetBeans: es un proyecto de código abierto de gran éxito con una gran base de usuarios y fácil de usar. La actual versión de este IDE (NetBeans 6.0.1) permite la integración con SVN. [16]

Eclipse es un IDE que emplea módulos (plug-in) para toda la gama de funcionalidades que posee, a diferencia de otros entornos rígidos donde las mismas aparecen prefijadas, sean de utilidad para el usuario o no. Esto permite también el soporte de otros lenguajes de programación y la introducción de otras aplicaciones auxiliares que pueden resultar útiles durante el proceso de desarrollo como: herramientas UML, editores visuales de interfaces, ayuda en línea para librerías; además de la gran ventaja de ser una herramienta open-source y multiplataforma. [16]

El IDE de desarrollo seleccionado para continuar la implementación de la aplicación fue NetBeans debido a que posee un potente editor de código y una interfaz amigable para realizar el control de versiones subversión.

Conclusiones

En el presente capítulo se explicaron los fundamentos teóricos de los Sistemas Biológicos y la estimación de parámetros, los cuales forman parte de la información que se necesita gestionar para simular Sistemas Biológicos. El sistema que se va a desarrollar gestiona de manera ágil y personalizada dicha información.

Se seleccionó para dar solución a la problemática planteada el uso de un nuevo estándar, Portlets debido a las ventajas que brinda. Para desarrollar dicho estándar, después de analizar las diferentes técnicas y herramientas, se seleccionó la metodología de desarrollo Open UP, el lenguaje de

modelado visual UML, la herramienta Case Visual Paradigm, el lenguaje de programación Java y finalmente como IDE de desarrollo el NetBeans.

CAPÍTULO 2: CARACTERISTICAS DEL SISTEMA

En este capítulo se define lo que debe hacer la aplicación a través de los requerimientos funcionales y no funcionales, se realiza el diagrama de caso de uso del sistema, así como las descripciones de los casos de uso del mismo.

2.1 Breve Descripción del Sistema Propuesto

El sistema a desarrollar constituirá una herramienta computacional para la modelación matemática de Sistemas Biológicos que puedan ser descritos mediante dinámica de población. Podrá funcionar sin la necesidad de usar sistemas externos pero a la vez, será diseñado de tal forma que pueda ser integrado con otras herramientas como, un editor de ecuaciones y otra para la simulación de los modelos de los Sistemas Biológicos. La aplicación debe permitir el acceso a los usuarios que estén autenticados a los diferentes servicios. Será uno de los resultados alcanzados por el grupo de Bioinformática de la Universidad de las Ciencias Informáticas.

2.2 Modelo de Dominio

La modelación del negocio es una de las fases que RUP incluye en el desarrollo de software, uno de sus objetivos es comprender el objeto a automatizar. Cuando no existen procesos definidos, RUP propone realizar un modelo del dominio, que es un subconjunto del modelo de negocio. La metodología OpenUp aunque no realiza necesariamente este modelo, también lo puede desarrollar por su característica de ser una metodología flexible.

Un modelo de dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las cosas que existen o eventos que ocurren en el entorno en el que trabaja el sistema [17]. El modelo de dominio se describe mediante diagramas UML (específicamente mediante diagramas de clases).

En la figura que aparece a continuación se representa como quedó estructurado el modelo de dominio del Sistema propuesto.

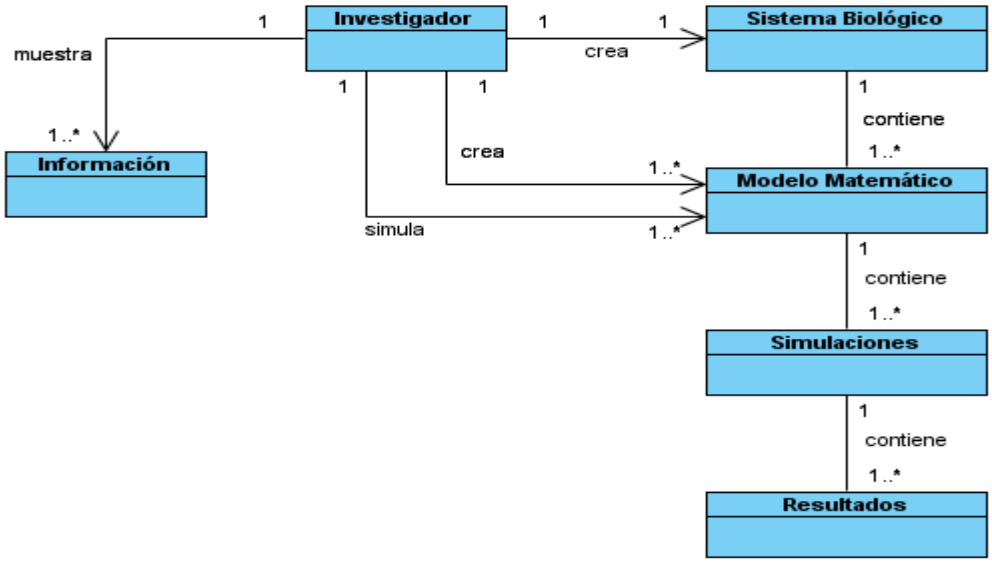


Fig.1 Modelo de Dominio del Sistema

2.2.1 Descripción textual del modelo de dominio

Un investigador es una persona que ejecuta acciones sistemáticas orientadas a la creación y generación de nuevo conocimiento.

Los Sistemas Biológicos son, a menudo, descritos como ecuaciones o como un conjunto de interacciones no lineales entre sus elementos. Son sistemas abiertos que operan en condiciones alejadas del equilibrio termodinámico.

Un modelo matemático se define como: una descripción desde el punto de vista de las matemáticas de un hecho o fenómeno del mundo real, desde el tamaño de la población, hasta fenómenos físicos como la velocidad, aceleración o densidad. El objetivo del modelo matemático es entender ampliamente el fenómeno y tal vez, predecir su comportamiento en el futuro.

La simulación se define como: una técnica numérica utilizada para representar un proceso o fenómeno mediante otro más simple que permite analizar sus características. Esta técnica emplea relaciones matemáticas y lógicas que son necesarias para describir el comportamiento y estructura de sistemas complejos del mundo real a través de periodos de tiempo prolongado.

Información: es un conjunto organizado de datos de las simulaciones realizadas, que constituye un

mensaje sobre el sistema biológico. Permitiendo resolver problemas y tomar decisiones, su uso racional es la base del conocimiento.

Resultado: es todo lo que se ha obtenido a partir de Gestionar Sistemas, Modelos Matemáticos y Simulaciones.

2.3 Actores del Sistema

Los actores de un sistema pueden ser otros sistemas o hardware externo que se relacionan o interactúan con dicho sistema, no necesariamente tiene que ser una persona. Cada actor juega un rol determinado al interactuar con el sistema y diferentes usuarios pueden asumir el mismo rol de un actor. [18]

Luego de definir el concepto de actores del sistema se puede establecer el o los actores del sistema en cuestión. A continuación aparece el actor del Sistema establecido.

Actor	Descripción
Investigador	El investigador es aquella persona que va a interactuar con el sistema para realizar la configuración, modelación matemática y las simulaciones de los Sistemas Biológicos, además de actualizar el idioma con en el que desee que aparezca la aplicación.

Fig.2 Actor del sistema

2.4. Definición de los Requisitos Funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir. [19] La aplicación a desarrollar a través de la realización del presente trabajo debe contar con los siguientes requisitos:

RF1.Gestionar sistema biológico.

RF1.1 Insertar sistema biológico.

RF1.2 Modificar sistema biológico.

RF1.3 Eliminar sistema biológico.

RF2.Gestionar modelos matemáticos.

RF2.1 Insertar modelo.

RF2.2 Modificar modelo.

RF2.3 Eliminar modelo.

RF3.Gestionar simulación.

RF3.1 Crear simulación.

RF3.2 Detener simulación.

RF3.3 Ver estado de la simulación.

2.4.1 Descripción de los Requisitos Funcionales

RF1.Gestionar sistema biológico.

Este requisito funcional, gestiona los Sistemas Biológicos al insertar, modificar o eliminar el nombre y la descripción de los mismos.

RF2.Gestionar modelos matemáticos.

Este requisito funcional, gestiona los modelos matemáticos al insertar, modificar o eliminar los modelos de cada sistema.

RF3.Gestionar simulación.

Este requisito funcional, gestiona la simulación al crear, detener o ver estado de la simulación de los modelos de cada sistema.

2.5 Definición de lo Requisitos No Funcionales

Los requisitos no funcionales son otros requisitos que no forman parte de la funcionalidad principal de la aplicación, como requisitos del entorno de desarrollo o ejecución (sistema operativo, servidores en los que correrá, lenguajes, etc.), restricciones que se aplicarán, prestaciones (tiempo de respuesta mínimo, alta disponibilidad, etc.), fiabilidad, portabilidad, interfaces externas, seguridad y otros. Se puede decir que responden a aquellas cualidades y características que el producto debe tener para que sea atractivo, confiable, usable y seguro. [19]

Dentro de los requisitos no funcionales para el desarrollo de la aplicación se encuentran:

Usabilidad

El sistema le ofrecerá al investigador la posibilidad de editar las ecuaciones del modelo, variar sus parámetros, realizar simulaciones distribuidas y analizar los resultados de las simulaciones. El diseño de la aplicación está centrado en ello y en específico en el diseño de las interfaces que harán posible el intercambio de datos de modo que resulte fácil el uso del sistema.

El sistema le ofrecerá al investigador la posibilidad de realizar simulaciones distribuidas, en este sentido se centra el diseño de la aplicación y en específico de las interfaces que harán posible el intercambio de datos.

La aplicación dará la posibilidad de almacenar los resultados obtenidos a partir de la simulación de Sistemas Biológicos, mediante interfaces que permitan el intercambio de datos de manera fácil para el usuario.

Se debe lograr un diseño adaptable, con la capacidad de poder soportar funcionalidades adicionales o modificar las funcionalidades existentes, sin impactar el resto de los requerimientos contemplados en el sistema.

Fiabilidad

Cada uno de los módulos del sistema debe proporcionar funcionalidades propias que satisfagan las necesidades establecidas y actuar bajo determinadas condiciones especificadas.

Confidencialidad

Se requiere de usuario y contraseña para poder acceder al Sistema.

Disponibilidad

En caso de tener usuario y contraseña, se le garantiza poder acceder a su información almacenada, en todo momento.

De existir algún fallo en el Sistema la información del usuario será guardada y protegida mientras se restablezcan las conexiones.

De ocurrir algún fallo en las conexiones para realizar los cálculos distribuidos, el sistema está diseñado para detectar este error y enviar la parte del trabajo que no se concluyó a otra máquina que esté disponible para que esta la continúe.

Integridad

La información manejada por el sistema será objeto de cuidadosa protección contra la corrupción y estados inconsistentes, de la misma forma, será considerada igual a la fuente o autoridad de los datos. Pueden incluir también mecanismos de chequeo de integridad y realización de auditorías.

Rendimiento

El sistema debe ser capaz de formular respuestas lo más rápido posible. Para hacer más rápida la obtención de los resultados de las simulaciones se hace uso de la plataforma de cálculo distribuido, posibilitando agilidad en la obtención de los resultados.

Soporte

El sistema estará bien documentado de forma tal que el tiempo de mantenimiento sea mínimo en caso de necesitarse.

El sistema permite implementar cambios, ya sea cualquier corrección, mejora o adaptación del software, por ejemplo: adicionar un nuevo módulo, sin efectos inesperados.

El sistema debe funcionar en cualquier sistema operativo sobre el cual se haya instalado un navegador Web.

Instalación: La instalación del sistema debe caracterizarse por su facilidad, claridad y sencillez (esto es para los que administren o instalen el sistema, no para los usuarios finales del mismo).

Portabilidad: El sistema debe ser multiplataforma (ser capaz o caracterizarse por poder funcionar o mantener una interoperabilidad de forma similar en diferentes sistemas operativos o plataformas).

El sistema debe permitir la interacción con los demás módulos que componen la plataforma.

Se realizarán distintas pruebas al software una vez concluido para comprobar su funcionalidad.

Terminado el software se prestarán los servicios de instalación y configuración de la aplicación.

Se prestarán servicios de mantenimiento del software.

El sistema será capaz de soportar los idiomas: inglés y español.

Interfaces del sistema

Interfaz externa amigable, legible, interactiva, fácil de usar, profesional, clara y sencilla.

El sistema deberá tener una interfaz externa fácil de entender por el investigador, que le brinde a éste una forma escalonada de acceder a los distintos métodos de análisis en interfaces diferentes, presentándole así una organización jerárquica evitando que el usuario pierda la orientación dentro de la aplicación.

El sistema no tendrá muchas imágenes y tendrá pocos colores.

Interfaz de usuario

Navegación y distribución: la distribución de las interfaces está en correspondencia con la actividad que el usuario vaya a realizar de manera coherente y lógica, al realizar cada uno de los pasos dentro de la investigación que haga en la aplicación.

Consistencia: las interfaces de trabajo tienen consistencia con la interfaz principal, manteniendo y relacionando correctamente los Sistemas Biológicos con las tareas de simulación y análisis que se realizan en las interfaces destinadas a este trabajo.

Personalización y adaptación al cliente: el cliente podrá ajustar tanto la interfaz principal como las

demás interfaces de trabajo a su gusto y facilidad.

Restricciones del sistema

- **Software**:

La máquina virtual de java tiene que estar instalada en su versión 1, 5 o superior.

El sistema operativo a instalar en la PC donde se ejecuta el sistema, sin restricciones, por ser un software multiplataforma.

El Sistema Operativo a instalar en las PC clientes del T-arenal con cualquier distribución de Linux.

Asistente matemático MatLab instalado en cada una de las máquinas de la plataforma distribuida, en caso de no estarlo se usará java.

Cliente del T-arenal instalado en cada una de las PCs que se utilizarán para la simulación distribuida.

- **Hardware:**

En el cliente:

Se requiere de 128 MB de memoria RAM.

Procesadores Pentium III.

Disco duro de 40 GB.

En el servidor:

Se requiere de 512 MB de memoria RAM.

Procesadores Pentium IV.

Disco duro de 40 GB (puede variar dependiendo de la cantidad de información a almacenar).

Servidor Web Apache Tomcat.

Restricciones en el diseño y la implementación.

Lenguaje de programación Java.

IDE de desarrollo NetBeans.

Herramienta CASE Visual Paradigm.

Gestor de bases de datos PostgreSQL.

Para el desarrollo del sistema se va a usar el Framework para Java Hibernate, que sigue de forma estricta el patrón DAO, se hará uso del mismo para el acceso a los datos.

El diseño de cada una de los componentes que conforman el sistema debe seguir el patrón en MVC (Modelo Vista Controlador).

Para la implementación de esta versión Web se reutilizará código implementado en la primera versión del mismo.

El Sistema de Cómputo Distribuido en Java como componente de proveedores externos, debe estar disponible siempre que sea solicitado su uso.

2.6 Patrones de Casos de Uso

Uno de los patrones utilizados para hacer más fácil el trabajo con los sistemas y mucho más simple su mantenimiento fue el patrón **CRUD** (Creating, Reading, Updating, Deleting).

CRUD consta de un caso de uso, llamado Información CRUD o Gestionar información, modela todas las operaciones que pueden ser realizadas sobre una parte de la información de un tipo específico, tales como: creación, lectura, actualización y eliminación. Suele ser utilizado cuando todos los flujos contribuyen al mismo valor del negocio y estos a su vez son cortos y simples. [20]

Listado de Casos de Uso.

CU-1 Gestionar sistema.

CU-2 Gestionar modelo matemático.

CU-3 Gestionar simulación.

2.7 Diagrama de Casos de Uso del Sistema

Cada forma en que los actores usan un sistema, se representa con un caso de uso y éstos son fragmentos de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores. De manera más precisa, un caso de uso especifica una secuencia de acciones que el sistema debe llevar a cabo, interactuando con sus actores dónde se obtiene un resultado de valor para los actores.

Un diagrama de casos de uso del sistema contiene actores, casos de uso del sistema y las relaciones existentes entre los mismos.

A continuación, se representa el diagrama de casos de uso definido para el Sistema, donde se muestra la relación entre el Investigador (actor) y los casos de uso Gestionar sistema biológico, Gestionar modelo matemático y Gestionar Simulación. Ver Fig.3

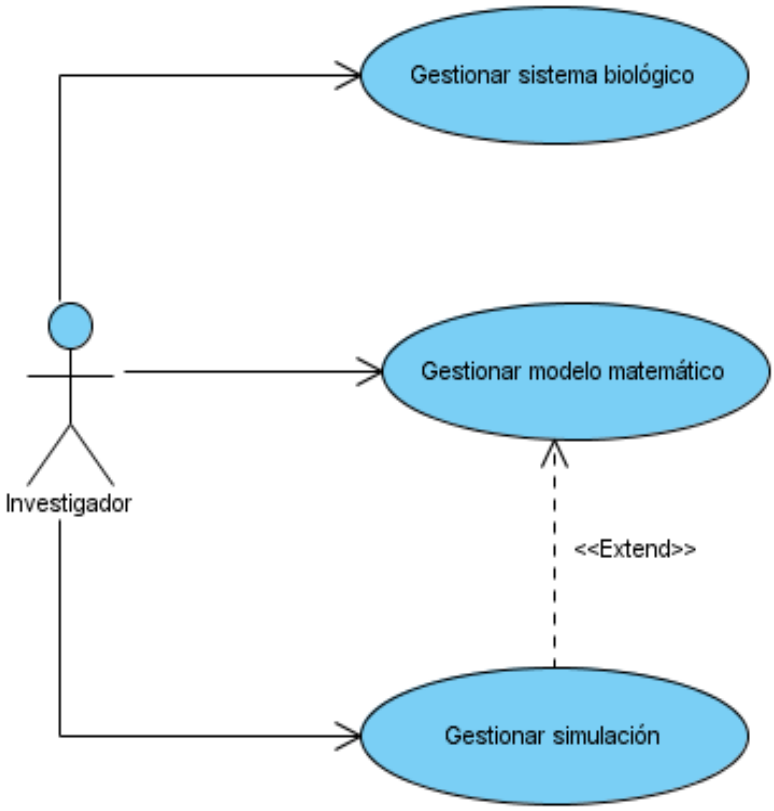


Fig.3 Diagrama de casos de uso del sistema

2.8 Descripción de Casos de Uso del Sistema

A continuación se hace una descripción de los casos de uso establecidos en el Sistema.

2.8.1 Descripción del Caso de Uso Gestionar sistema biológico.

El caso de uso Gestionar sistema biológico: permite al investigador Insertar, Modificar o Eliminar Sistemas Biológicos. El caso de uso inicia cuando el investigador decide realizar algunas de estas acciones y se dirige al menú seleccionando la opción Gestionar sistema, seguidamente el sistema le muestra una interfaz donde puede realizar dichas operaciones.

Si el investigador necesita Insertar un nuevo Sistema Biológico, en la sección de Insertar de la interfaz mostrada el investigador inserta el nombre y la descripción textual, el sistema verifica que el nombre y/o la descripción no estén en blanco, verificando además, que el nombre no esté registrado en la aplicación. Luego, si todos los datos introducidos están correctos lo guarda en la base de datos y finalmente el sistema notifica al investigador que se insertó satisfactoriamente. En caso contrario, el sistema muestra los mensajes pertinentes.

Si el investigador necesita Modificar un Sistema Biológico, en la interfaz mostrada, selecciona la opción “Editar” del Sistema Biológico que desea modificar, el sistema muestra una interfaz con el nombre y la descripción del Sistema Biológico seleccionado. Seguidamente el Investigador modifica el nombre y/o la descripción. El sistema verifica que el nombre y/o la descripción no estén en blanco, verifica además que el nombre no esté registrado en la aplicación, luego si todos los datos modificados están correctos los guarda en la base de datos y finalmente el sistema notifica al investigador de que se modificó satisfactoriamente. En caso contrario, el sistema muestra los mensajes pertinentes.

Si el investigador necesita Eliminar un Sistema Biológico, en la interfaz mostrada, selecciona la opción “Eliminar” del Sistema Biológico que desea eliminar, el sistema elimina el Sistema Biológico de la aplicación. En caso contrario, el sistema muestra el mensaje correspondiente. Ver Fig.4

Caso de Uso:	Gestionar sistema biológico
Actores:	Investigador

Resumen:	El caso de uso inicia cuando el Investigador decide Insertar Sistema Biológico, Modificar un Sistema Biológico o Eliminar un Sistema Biológico.
Referencia:	**RF1, RF1.1, RF1.2, RF1.3.**
Prioridad:	Crítico
Precondiciones:	El investigador debe estar autenticado en el sistema
Flujo Normal de Eventos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Investigador del sistema se dirige al menú y selecciona la opción Gestionar sistema.	1.1- El sistema brinda la posibilidad de realizar las operaciones: Insertar Sistema, Modificar Sistema y Eliminar Sistema.
2- El Investigador decide la operación a realizar.	2.1- Si decide "Insertar Sistema" ver sección **Insertar Sistema.** - Si decide "Modificar Sistema" ver sección **Modificar Sistema.** - Si decide "Eliminar Sistema" ver sección **Eliminar Sistema.**
Sección "Insertar Sistema"	
3- El investigador inserta el nombre del Sistema Biológico y su descripción textual.	3.1-El sistema verifica que el nombre y/o la descripción del Sistema Biológico no estén en blanco.
	3.2-El sistema verifica que el nombre del Sistema Biológico no esté almacenado en la aplicación.
	3.3-Si todos los datos introducidos están correctos los almacena en la aplicación.
	3.4- El sistema notifica al investigador de que se insertó satisfactoriamente y finaliza el caso de uso.
Sección "Modificar Sistema"	
3- El Investigador selecciona el Sistema Biológico que desea modificar.	3.1- El sistema brinda la posibilidad de modificar los datos del Sistema Biológico seleccionado.

4- El Investigador modifica el nombre y/o la descripción del Sistema Biológico.	4.1- El sistema verifica que el nombre y/o la descripción del Sistema Biológico no estén en blanco.
	4.2 El sistema verifica que el nombre del Sistema Biológico no coincida con otro que esté registrado en la aplicación.
	4.3-Si todos los datos modificados están correctos, los guarda en la aplicación.
	4.4- - El sistema brinda la posibilidad de realizar las operaciones: Insertar Sistema, Modificar Sistema y Eliminar Sistema, notificando al investigador de que se modificó satisfactoriamente y finaliza el caso de uso.
Sección "Eliminar Sistema"	
3- El Investigador selecciona el Sistema Biológico que desea eliminar.	3.1- El sistema elimina el Sistema Biológico seleccionado, de la aplicación.
	3.2- El sistema notifica al investigador de que se eliminó satisfactoriamente y finaliza el caso de uso.
Flujos Alternos	
Sección "Insertar Sistema"	
Acción del Actor	**Respuesta del Sistema**
3- El Investigador deja en blanco el nombre y/o la descripción del Sistema Biológico.	3.1- El sistema muestra un mensaje "el nombre y/o la descripción del Sistema Biológico está en blanco".
	3.2- Si el nombre del Sistema Biológico coincide con alguno que ya este registrado en la aplicación muestra un mensaje "El nombre de ese sistema ya existe".
	3.3- Si la aplicación no puede almacenar los datos, el sistema muestra un mensaje "No se pueden almacenar los datos en la aplicación".
Sección "Modificar Sistema"	

4- El Investigador deja en blanco el nombre y/o la descripción del Sistema Biológico.	4.1- El sistema muestra un mensaje "el nombre y/o la descripción del Sistema Biológico está en blanco".
	4.2- Si el nombre del Sistema Biológico coincide con alguno que ya este registrado en la aplicación, muestra un mensaje "El nombre de ese sistema ya existe".
	4.3- Si la aplicación no puede almacenar los datos, el sistema muestra un mensaje "No se pueden almacenar los datos en la aplicación".
Sección "Eliminar Sistema"	
	3.1- Si la aplicación no puede almacenar los datos, el sistema muestra un mensaje "No se pueden almacenar los datos en la aplicación".
Postcondiciones:	Se inserta un Sistema Biológico, se modifican los datos o se elimina el mismo. La aplicación tendrá guardada la información de los Sistemas Biológicos.

Fig.4 Descripción del caso de uso Gestionar sistema

2.8.2 Descripción del Caso de Uso Gestionar modelo matemático.

El caso de uso Gestionar modelo matemático permite al investigador Insertar, Modificar o Eliminar Modelos Matemáticos. El caso de uso inicia cuando el investigador decide realizar algunas de estas acciones y se dirige al menú seleccionando la opción Gestionar modelo. Seguidamente el sistema le muestra una interfaz donde aparecen todos los Sistemas Biológicos registrados en la aplicación. El investigador selecciona el sistema al que desea gestionarle un modelo. Posteriormente el sistema muestra todos los modelos pertenecientes al Sistema Biológico seleccionado, donde el usuario puede decidir la opción a realizar.

Si el investigador opta por la opción Insertar un nuevo modelo, el sistema le muestra la herramienta Editor de Ecuaciones donde el investigador inserta el nombre del modelo y la ecuación diferencial, el sistema verifica que el nombre no esté en blanco, luego si los datos introducidos están correctos los guarda en la base de datos y finalmente el sistema notifica al investigador de que se insertó

satisfactoriamente. En caso contrario, el sistema muestra los mensajes pertinentes.

Si el investigador necesita Modificar un modelo, en la interfaz mostrada, selecciona la opción “Editar” del modelo que desea modificar, el sistema verifica que el modelo seleccionado no haya sido simulado, consecutivamente muestra la herramienta Editor de Ecuaciones con los datos del modelo seleccionado, seguidamente el Investigador modifica la ecuación diferencial. Luego si todos los datos introducidos están correctos lo guarda en la base de datos y finalmente el sistema notifica al investigador de que se modificó satisfactoriamente. En caso contrario, el sistema muestra los mensajes pertinentes.

Si el investigador necesita Eliminar un modelo, en la interfaz mostrada, selecciona la opción “Eliminar” del modelo que desea eliminar, el sistema elimina el modelo de la aplicación. En caso contrario, el sistema muestra el mensaje correspondiente.

Si por el contrario de Gestionar modelo, el investigador necesita Simular, en la interfaz mostrada, selecciona la opción “Simular”, el sistema le muestra todas las variables correspondientes al modelo seleccionado, dándole la posibilidad al investigador de introducir los datos necesarios para la simulación. Luego de introducir la información, el sistema le da la posibilidad de realizar la simulación y guardarla en la aplicación. Ver Fig.5

Caso de Uso:	Gestionar modelo matemático
Actores:	Investigador
Resumen:	El caso de uso inicia cuando el Investigador decide Insertar un Modelo Matemático, Modificar un Modelo Matemático o Eliminar un Modelo Matemático.
Referencia:	**RF2, RF2.1, RF2.2, RF2.3.**
Prioridad:	Crítico
CU asociados:	Gestionar simulación
Precondiciones:	El investigador debe estar autenticado en el sistema
Flujo Normal de Eventos	
Sección “General”	
Acción del Actor	**Respuesta del Sistema**

1- El Investigador del sistema se dirige al menú y selecciona la opción Gestionar modelo.	1.1- El sistema brinda la posibilidad de mostrar todos los Sistemas Biológicos registrados en la aplicación con su descripción textual.
2- El Investigador selecciona el Sistema Biológico al que desea Gestionar Modelo Matemático.	2.1- El sistema muestra los modelos matemáticos del Sistema Biológico seleccionado.
3- El Investigador selecciona la opción a realizar.	3.1- Si selecciona "Insertar Modelo" ver sección **Insertar Modelo.** - Si selecciona "Modificar Modelo" ver sección **Modificar Modelo.** - Si selecciona "Eliminar Modelo" ver sección **Eliminar Modelo.**
Sección "Insertar Modelo"	
	4- El sistema brinda la posibilidad de utilizar la aplicación: Editor de ecuaciones.
5- El Investigador introduce los datos deseados.	5.1- El sistema verifica que el nombre del Modelo Matemático no estén en blanco.
	5.2- El sistema guarda el modelo.
	5.3- El sistema notifica al investigador de que se insertó satisfactoriamente y finaliza el caso de uso.
Sección "Modificar Modelo"	
	4- El sistema verifica que el modelo seleccionado no haya sido simulado.
	4.1- El sistema brinda la posibilidad de utilizar el Editor de ecuaciones con los datos correspondientes al modelo seleccionado.
5- El Investigador modifica la ecuación diferencial del Modelo Matemático.	5.1- El sistema registra la información.
	5.2- El sistema notifica al investigador de que se modificó satisfactoriamente y finaliza el caso de uso.
Sección "Eliminar Modelo"	

	4.1- El sistema elimina el Modelo Matemático seleccionado, de la aplicación.
	4.2- El sistema notifica al investigador de que se eliminó satisfactoriamente y finaliza el caso de uso.
Flujos Alternos	
Sección "Insertar Modelo"	
Acción del Actor	**Respuesta del Sistema**
	5.1- Si el sistema verifica que el nombre del Modelo Matemático está en blanco, muestra un mensaje "El nombre del modelo está en blanco".
	5.2- Si la aplicación no puede almacenar los datos, el sistema muestra un mensaje "No se pueden almacenar los datos en la aplicación".
Sección "Modificar Modelo"	
	4- Si el sistema verifica que el modelo seleccionado ya ha sido simulado o se está simulando, muestra un mensaje "Este modelo no se puede modificar."
	5.1- Si la aplicación no puede almacenar los datos, el sistema muestra un mensaje "No se pueden almacenar los datos en la aplicación".
Sección "Eliminar Modelo"	
	4.1- Si la aplicación no puede almacenar los datos, el sistema muestra un mensaje "No se pueden almacenar los datos en la aplicación".
Flujos Alternos	
3- El Investigador selecciona la opción simular.	3.1- El sistema brinda la posibilidad de mostrar todas las variables del modelo seleccionado a simular, dándole la posibilidad al Investigador de introducir los datos necesarios para la simulación.

4- El Investigador introduce los datos.	4.1- El sistema le brinda la posibilidad de hacer la simulación y la guarda en la aplicación.
	4.2 Si la aplicación encuentra algún problema cuando va a simular, le envía al usuario el mensaje correspondiente.
Postcondiciones:	Se inserta un Modelo Matemático, se modifican los modelos o se elimina el mismo. La aplicación tendrá guardada la información de los Modelos Matemáticos.

Fig.5 Descripción del caso de uso Gestionar modelo matemático

2.8.3 Descripción del Caso de Uso Gestionar simulación.

El caso de uso Gestionar simulación permite al investigador Crear, Detener o Ver estado de la Simulación. El caso de uso inicia cuando el investigador decide realizar algunas de estas acciones y se dirige al menú seleccionando la opción Gestionar simulación. Seguidamente el sistema muestra una interfaz donde aparecen todas las simulaciones registradas en la aplicación, el investigador selecciona la opción a realizar.

Si el investigador opta por la opción detener, el sistema elimina la simulación de la aplicación. Si el investigador opta por la opción Ver estado, el sistema muestra una breve descripción de la simulación seleccionada. En caso contrario, el sistema muestra el mensaje correspondiente. Ver Fig.6

Caso de Uso:	Gestionar simulación
Actores:	Investigador
Resumen:	El caso de uso inicia cuando el Investigador decide Crear una Simulación, Detener Simulación o Ver estado de la Simulación.
Referencia:	**RF3, RF3.1, RF3.2, RF3.3.**
Prioridad:	Crítico
CU asociados:	Gestionar modelo matemático
Precondiciones:	El investigador debe estar autenticado en el sistema
Flujo Normal de Eventos	

Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Investigador del sistema se dirige al menú y selecciona la opción Simulaciones.	1.1- El sistema brinda la posibilidad de mostrar todas las simulaciones registradas en la aplicación.
2- El Investigador selecciona la opción a realizar.	2.1- Si selecciona la opción "Detener Simulación" ver sección **Detener Simulación.** - Si selecciona "Ver Estado de la Simulación" ver sección **Ver Estado de la simulación.**
Sección "Detener Simulación"	
3- El Investigador selecciona la Simulación que desea detener.	3.1- El sistema elimina la Simulación de la aplicación.
	3.2- El sistema notifica al usuario "La simulación ha sido cancelada." Y finaliza el caso de uso.
Sección "Ver Estado de la Simulación"	
3- El Investigador selecciona la Simulación a la que desea ver el estado.	3.1- El sistema muestra al Investigador una explicación detallada del estado de la Simulación seleccionada y finaliza el caso de uso.
Postcondiciones:	Se crea una Simulación, se ve el estado de la Simulación o se detiene la misma. La aplicación tendrá guardada la información de las simulaciones.

Fig.6 Descripción del caso de uso Gestionar simulación

Conclusiones

En el capítulo presentado se explica cómo se elaboraron artefactos requeridos por el Open Up para el desarrollo del software, funcionalidades que debe realizar el software representada por los requisitos funcionales y las cualidades que él mismo debe tener en los requisitos no funcionales, actores que interactúan con el sistema y el diagrama con una descripción detallada de los casos de uso del sistema.

CAPITULO 3: DISEÑO DEL SISTEMA

En este capítulo, se traducen los requisitos a una especificación que describe como implementar el sistema a través del diseño. Se realizan los diagramas de interacción y los diagramas de clases del diseño según los casos de uso definidos anteriormente. Se explica la arquitectura utilizada y los patrones de diseño empleados en la aplicación. Además, se realiza el modelo de despliegue que define como quedaría en general el sistema.

3.1 Arquitectura del sistema

3.1.1 Estilo arquitectónico utilizado

Para desarrollar la aplicación web se utilizó el patrón arquitectónico Modelo Vista Controlador:

- La Vista: es la información presentada al usuario y está formada por JSPs, que son las páginas con las que interactúa directamente el usuario desde la aplicación.
- El Controlador, actúa como intermediario entre el Modelo, la Vista y cualquier otro recurso necesario para generar una página.
- El modelo de datos, puede ser implementado a través de componentes que representan objetos en la base de datos.

A continuación se muestra este patrón de arquitectura. Ver Fig. 7

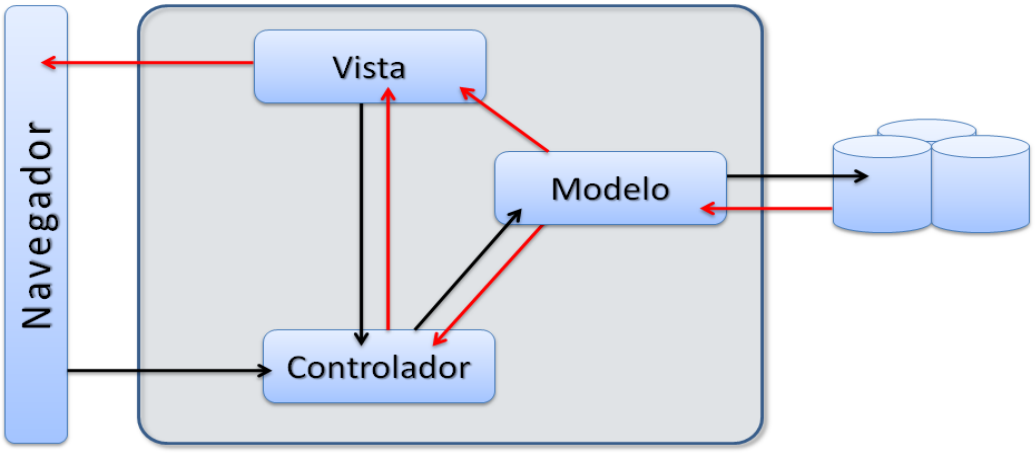


Fig.7 Modelo Vista Controlador

3.1.2 Arquitectura de Portlet

Se implementa el patrón de arquitectura MCV (Model View Controller) para el desarrollo de portlet. Contiene un Objeto implementado como Portlet parametrizable desde un archivo XML: el responsable de controlar el flujo de la aplicación, instanciar los beans que sean necesarios y servir los JSP (Vista) que estén activos en cada área. Controla los estados por los que pasa la aplicación desde que un usuario genera un evento, hasta que se devuelve el contenedor asociado. [3] En la capa de datos se almacena, de forma persistente, toda la información necesaria para facilitar los servicios ofrecidos por la aplicación. BSDAO.jar es el paquete que contiene todas las clases persistentes para la conexión.

3.1.3 Vista Lógica

Diagrama de clases de Alto Nivel

A continuación se pone de manifiesto los tres casos de uso que son programados en nuestra aplicación, los cuales son, a su vez, casos de uso arquitectónicamente significativos. Ver Fig. 8.

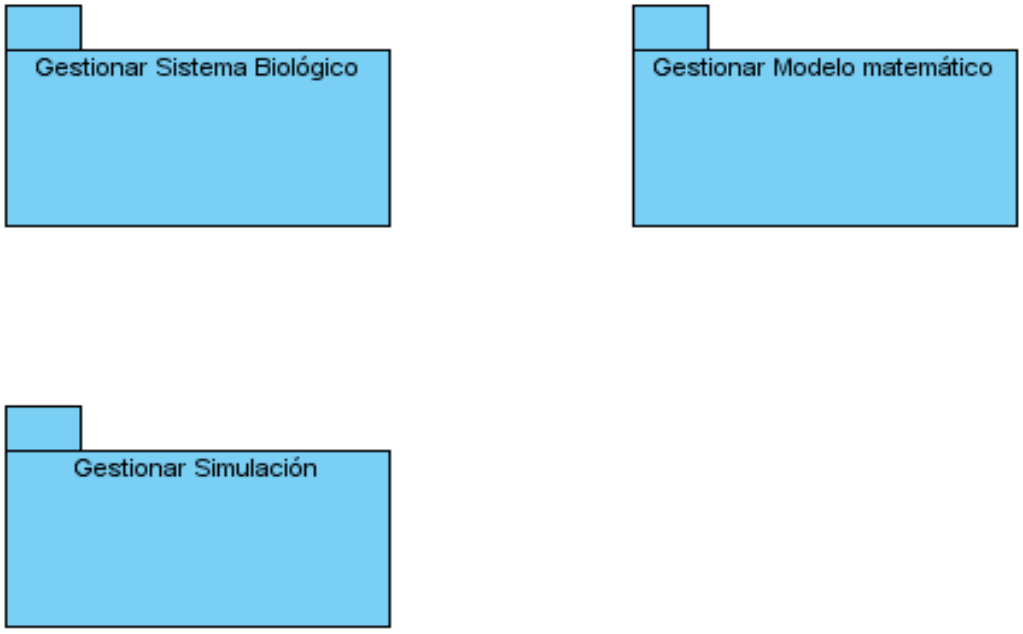


Fig.8 Diagrama de paquetes del diseño.

3.2 Diagrama de Interacción (secuencia)

Estos diagramas muestran una interacción, que consiste en un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar entre ellos. Los diagramas de interacción se utilizan para modelar los aspectos dinámicos del sistema. Se clasifican en Diagramas de Secuencia y Diagramas de Colaboración, los primeros destacan la ordenación temporal de los mensajes y los segundos la organización estructural de los objetos que envían y reciben mensajes. [21]

A continuación se muestran los Diagramas de Secuencia correspondientes a los Casos de Uso Gestionar sistema biológico, Gestionar modelo matemático y Gestionar simulación, ver Fig. 9, Fig. 10 y Fig. 11 respectivamente.

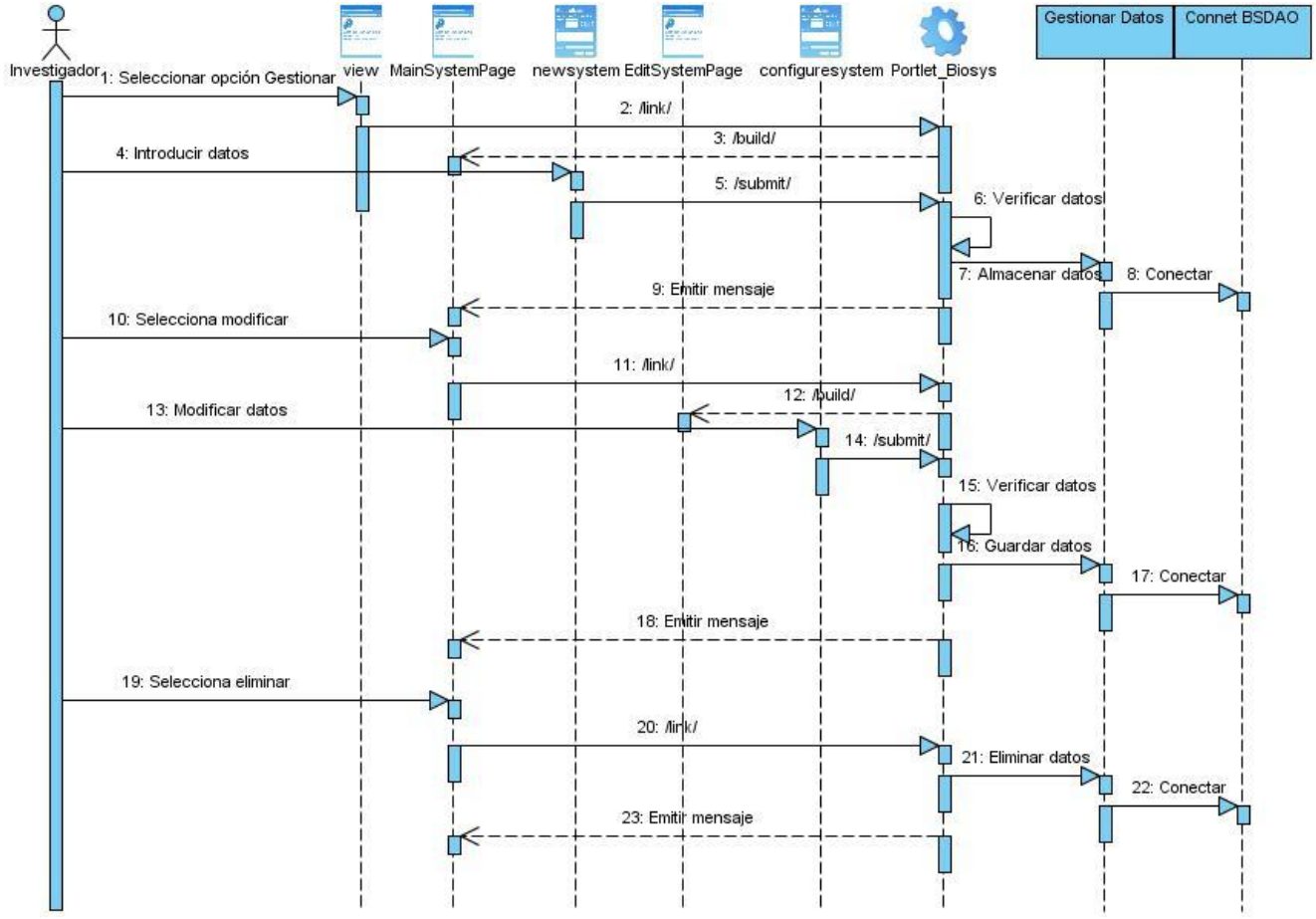


Fig.9 Diagrama de Secuencia del CU_Gestionar sistema biológico

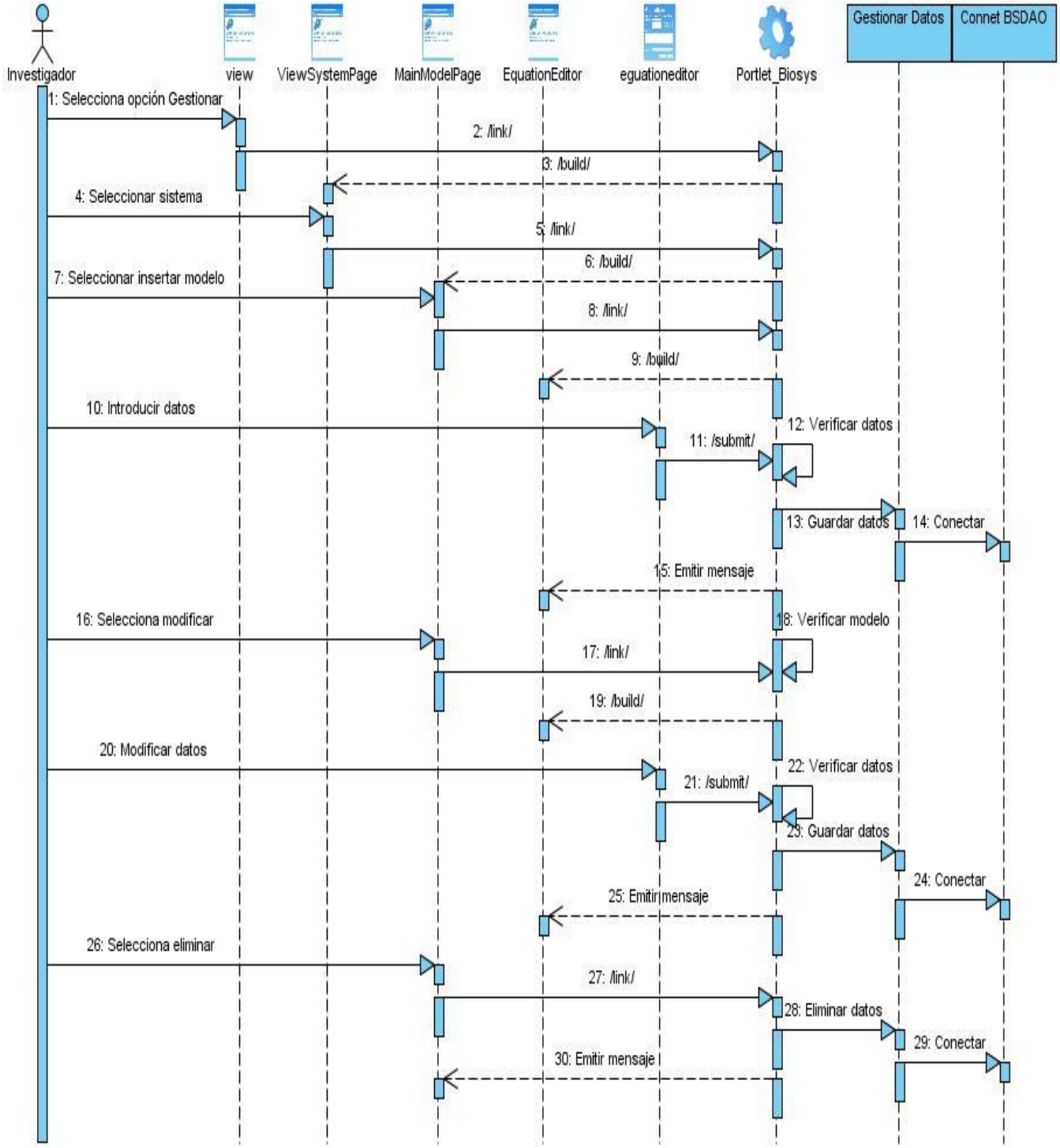


Fig. 10 Diagrama de Secuencia del CU_Gestionar modelo matemático

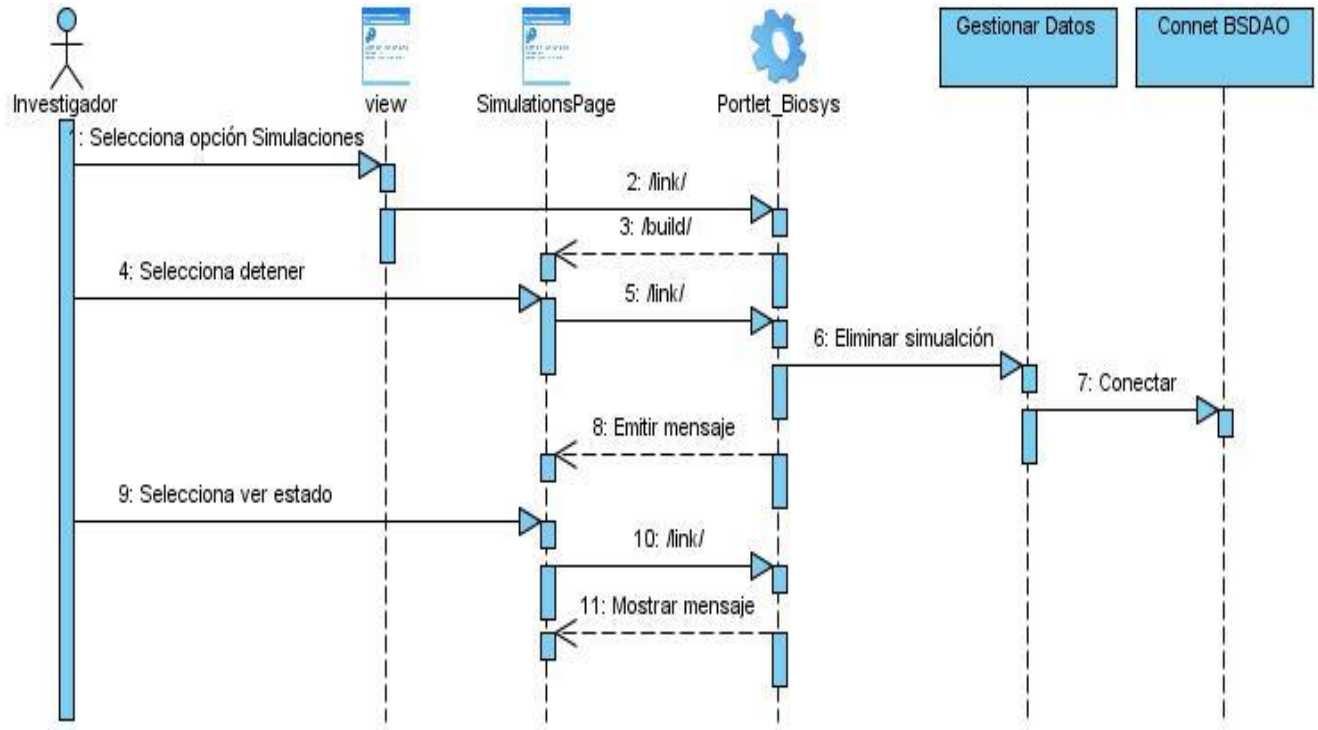


Fig. 11 Diagrama de Secuencia del CU_Gestionar simulación

3.2.1 Diagramas de secuencias de Flujos Alternos

Flujo alterno: acción inversamente proporcional al flujo normal de eventos.

En los siguientes anexos se representan los flujos alternos de cada uno de los diagramas de secuencia (flujo normal de eventos), de los casos de usos Gestionar sistema biológico, Gestionar modelo matemático y Gestionar simulación respectivamente.

Ver anexo 1

Ver anexo 2

Ver anexo 3

3.3 Patrones de diseño utilizados

Los patrones de diseño son soluciones estándares a problemas comunes en el desarrollo de software. Sirven como estructura, como soporte para el desarrollo de un sistema. [3]

3.3.1 Patrones GRASP

3.3.1.1Patrón Creador

El patrón creador nos ayuda a identificar quién debe ser el responsable de la creación de nuevas clases u objetos. [22]

Problema: ¿Quién debería ser responsable de crear una nueva instancia de alguna clase?

Solución: Asignarle a la clase B la responsabilidad de crear una instancia de clase A en uno de los siguientes casos:

- B agrega los objetos A.
- B contiene los objetos A.
- B registra las instancias de los objetos A.
- B utiliza especialmente los objetos A.
- B tiene los datos de inicialización que serán transmitidos a A cuando este objeto sea creado (así que B es un Experto respecto a la creación de A). B es un creador de los objetos A. [23]

Aplicación: Las clases EditorServlet y Portlet_Biosys**,** son responsables de crear instancias de la clase BSDAOFactory para realizar las funciones de insertar, modificar y eliminar datos en la base de datos.

3.3.1.2 Patrón Controlador

Problema: ¿Quién debería encargarse de atender un evento del sistema?
Solución: Asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase que represente una de las siguientes opciones:

- El “sistema” global (controlador de fachada).
- La empresa u organización global (controlador de fachada).

- Algo en el mundo real que es activo (por ejemplo, el papel de una persona) y que pueda participar en la tarea (controlador de tareas).
- Un manejador artificial de todos los eventos del sistema de un caso de uso, generalmente denominados “Manejador<NombreCasodeUso>” (controlador de casos de uso). [23]

Aplicación: En el sistema se cuenta con la clase Portlet_Biosys que se encarga de manejar los eventos y la lógica del sistema.

3.3.1.3 Patrón Experto

Problema: ¿Quién debiera ser el responsable de conocer la información?
Solución: Asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad.

Aplicación: el sistema implementa este patrón en forma de servicios. Ejemplo de su aplicación: la clase EditorServlet y Portlet_Biosys al necesitar datos de los Sistemas biológicos como nombre y descripción, los gestiona a través de la clase BSDAOFactory, la cual es la encargada del acceso a datos.

3.3.1.4 Alta Cohesión

Problema: ¿Cómo mantener la complejidad dentro de límites manejables?

Solución: Asignar una responsabilidad de modo que la cohesión siga siendo alta.

Aplicación: En el sistema se implementa la clase EditorServlet, además de la clase Portlet_Biosys para distribuir la responsabilidad, de modo que no estuviera sobrecargada y favorecer la reutilización del código.

3.4 Diagrama de clases

El diagrama de clases de diseño describe gráficamente las especificaciones de las clases de software y de las interfaces en una aplicación.

UML no define concretamente un artefacto denominado "diagrama de clases del diseño", sino que lo identifica con un término más genérico: "diagrama de clases". [21]

A continuación se reflejan los Diagramas de clases correspondientes al diseño de los Casos de Uso Gestionar sistema biológico, Gestionar modelo matemático y Gestionar simulación, Ver Fig. 12, Fig. 13 y Fig. 14 respectivamente.

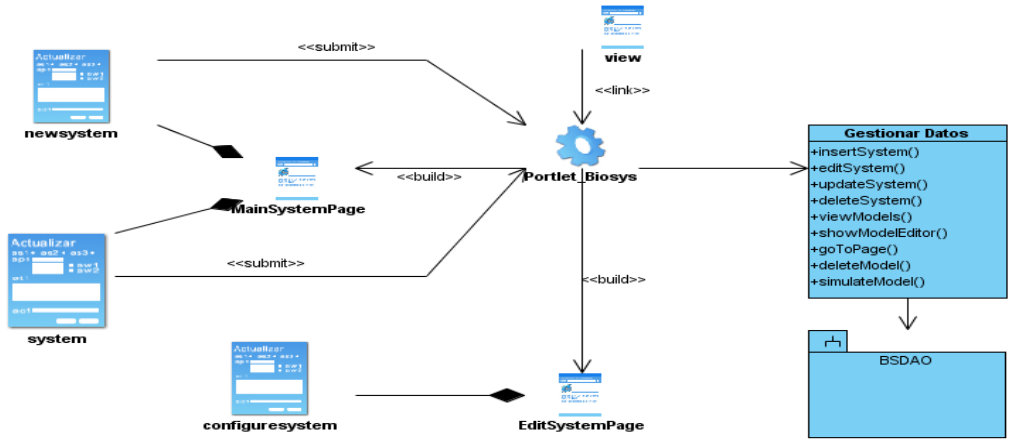


Fig. 12 CU_Gestionar sistema biológico

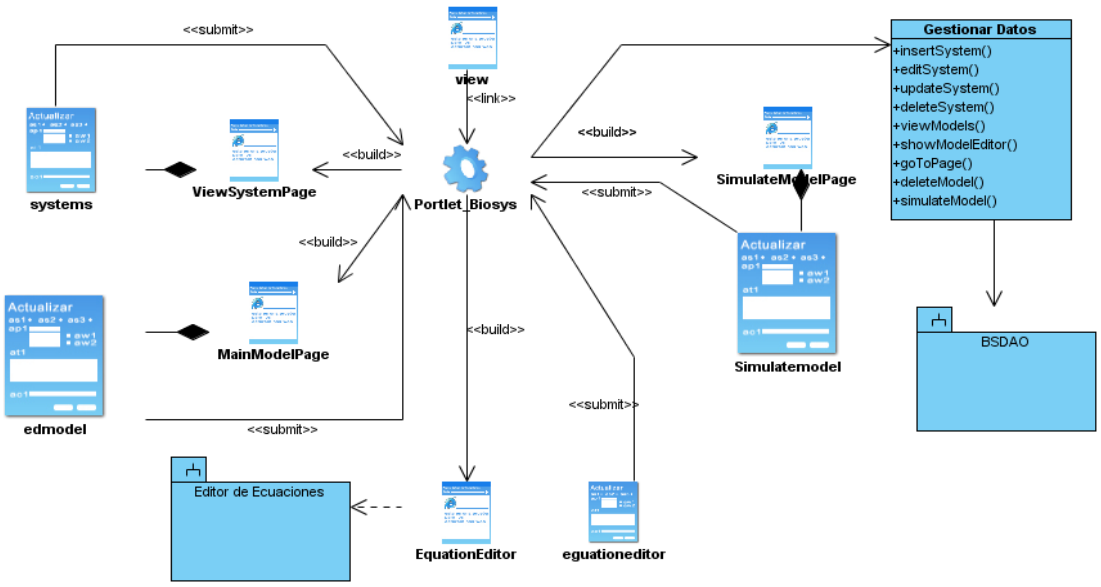


Fig.13 CU_Gestionar modelo matemático

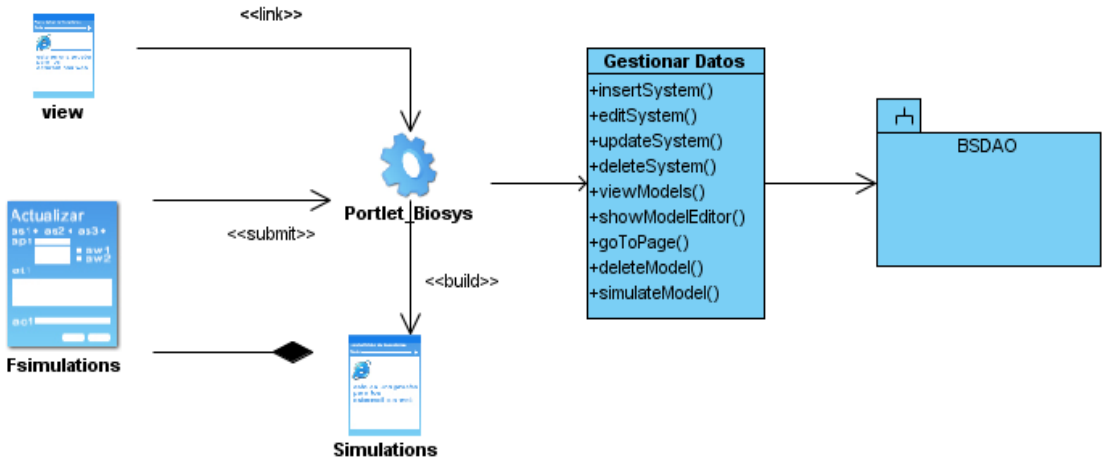


Fig. 14 CU_Gestionar simulación

Conclusiones

Como resultado de este capítulo se representan los diagramas de interacción y los diagramas de clases del diseño. Se hace una explicación de forma general del diseño, además de la fundamentación del uso de los patrones de diseño y arquitectura utilizados.

CAPITULO 4: IMPLEMENTACION Y PRUEBA

El objetivo principal de este capítulo es convertir los elementos del diseño en elementos de implementación. Se describe cómo los elementos del modelo del diseño se implementan en términos de componentes. También se presenta el código fuente de las principales clases y pantallas de la aplicación, por ejemplo: prototipos de interfaz de usuario y el mapa de navegación. Además, se exponen los resultados de las pruebas de aceptación desarrolladas, utilizando el método de caja negra para garantizar que el mismo cumpla con los requisitos funcionales.

4.1 Diagrama de Componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Los componentes representan todos los tipos de elementos software que entran en la fabricación de aplicaciones informáticas. Pueden ser simples archivos, paquetes, bibliotecas cargadas dinámicamente, etc. [23]

A continuación se muestra el diagrama de componentes perteneciente al sistema propuesto. Ver Fig. 15

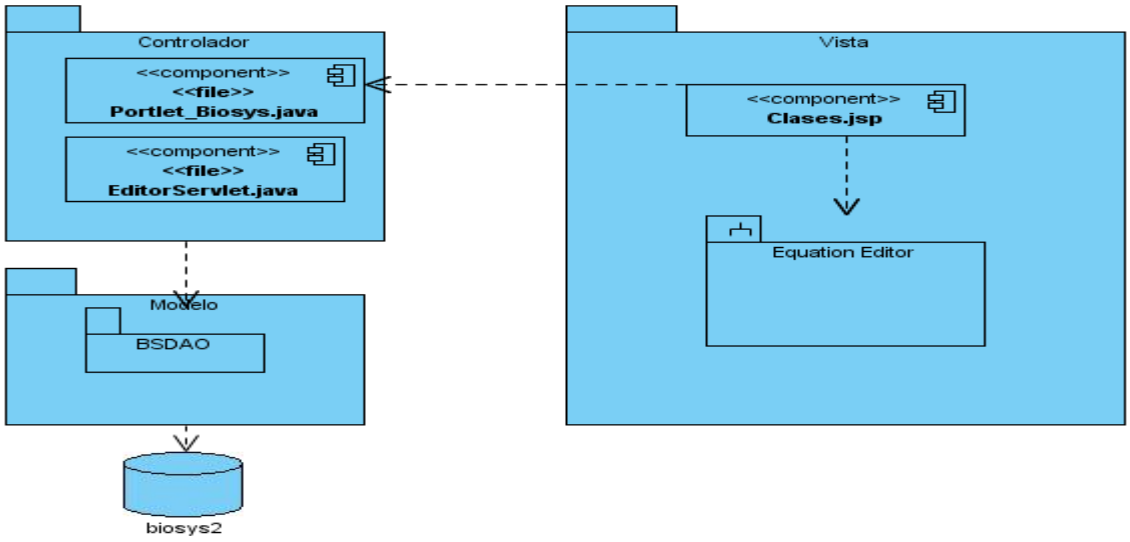


Fig. 15 Diagrama de componentes

4.2 Vista de Despliegue

Diagrama de Despliegue

Los diagramas de despliegue muestran la disposición física de los distintos nodos que entran en la composición de un sistema y el reparto de los programas ejecutables sobre estos nodos. [23]

A continuación se muestra la configuración hardware del sistema y los nodos físicos que la componen. Ver Fig. 16.

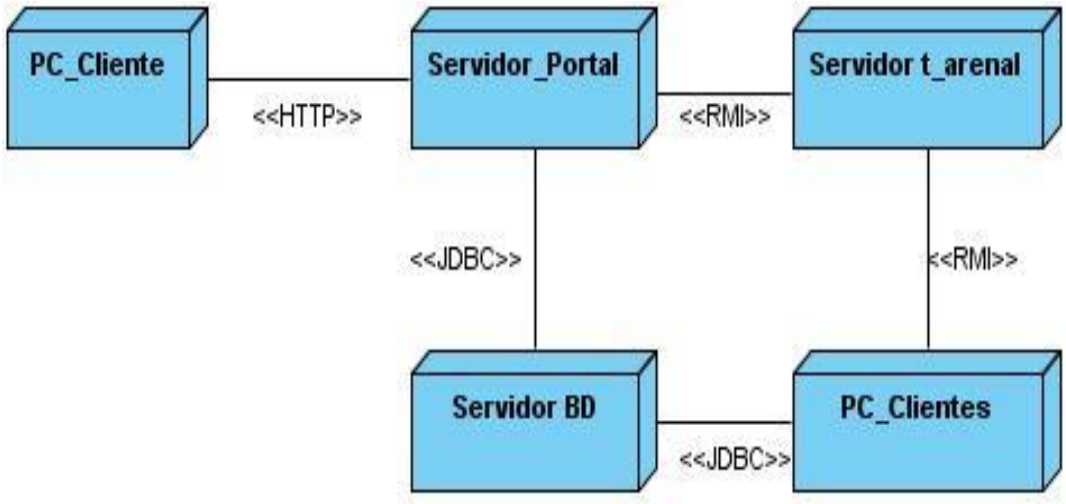


Fig. 16 Diagrama de Despliegue.

4.3 Mapa de Navegación

El modelo de navegación, se presenta en diferentes niveles que maneja el usuario al ingresar a cualquiera de las opciones expuestas. [24]

La figura expuesta a continuación, representa el mapa de navegación empleado en el sistema propuesto.

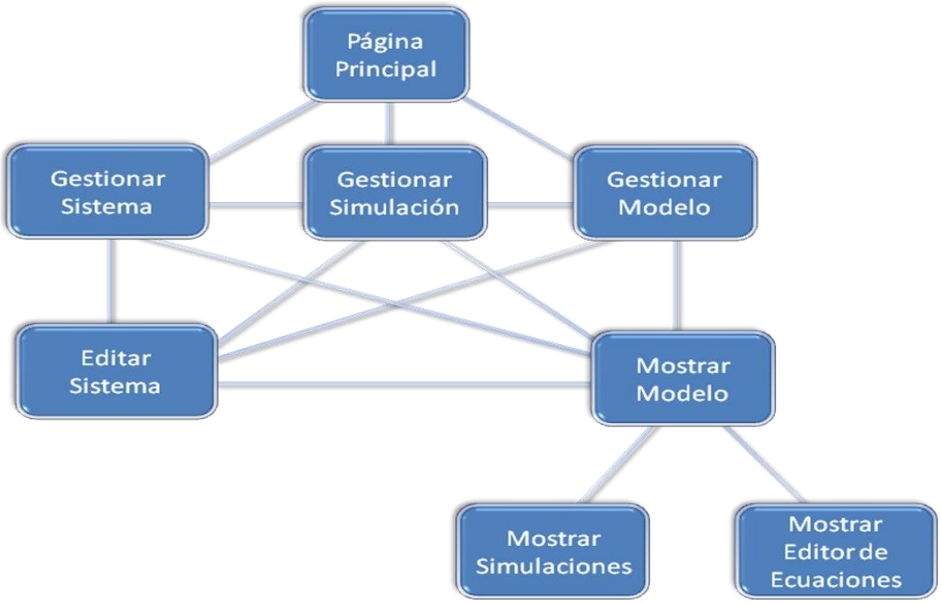
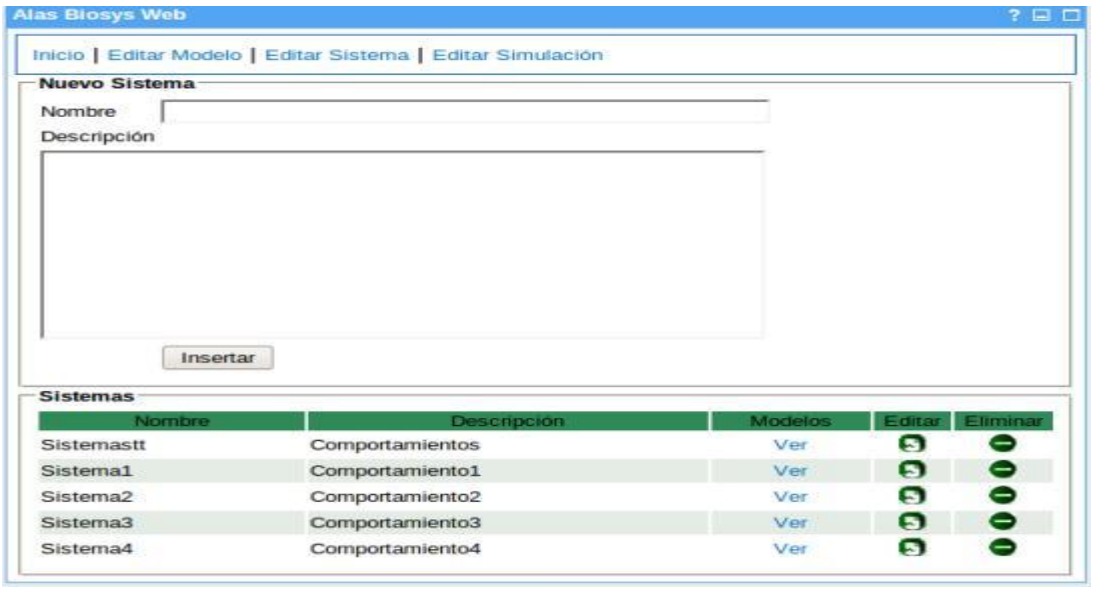


Fig. 17 Mapa de navegación

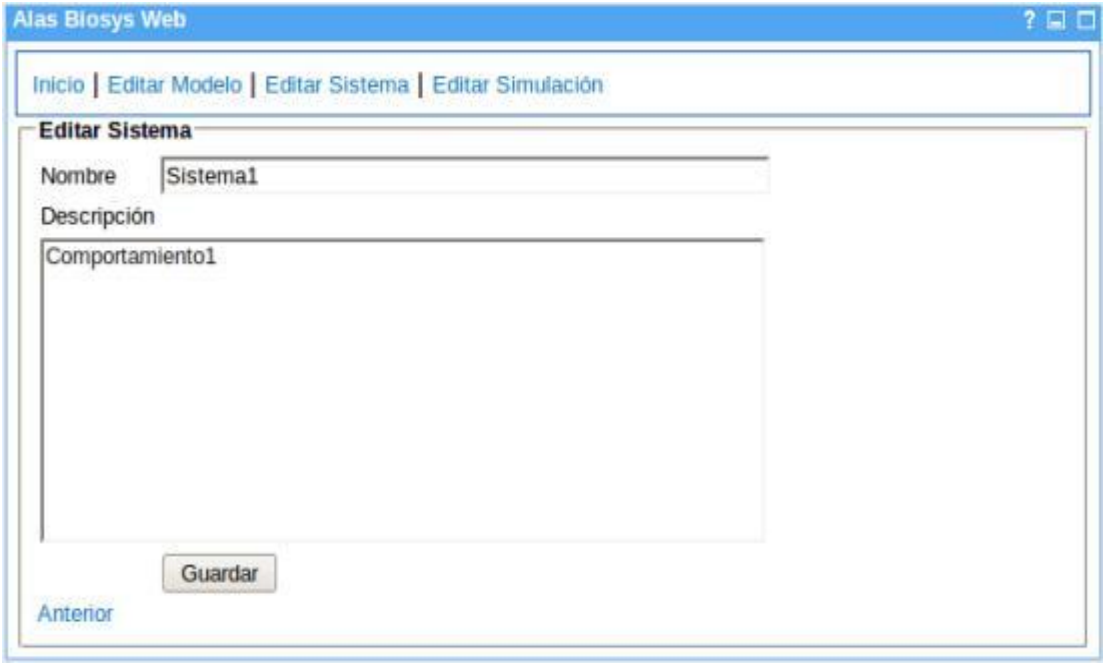
4.4 Pantallas de la Aplicación

La siguiente imagen refleja la página principal de la aplicación desarrollada.

La imagen siguiente, representa la interfaz correspondiente a la Gestión de sistemas biológicos.



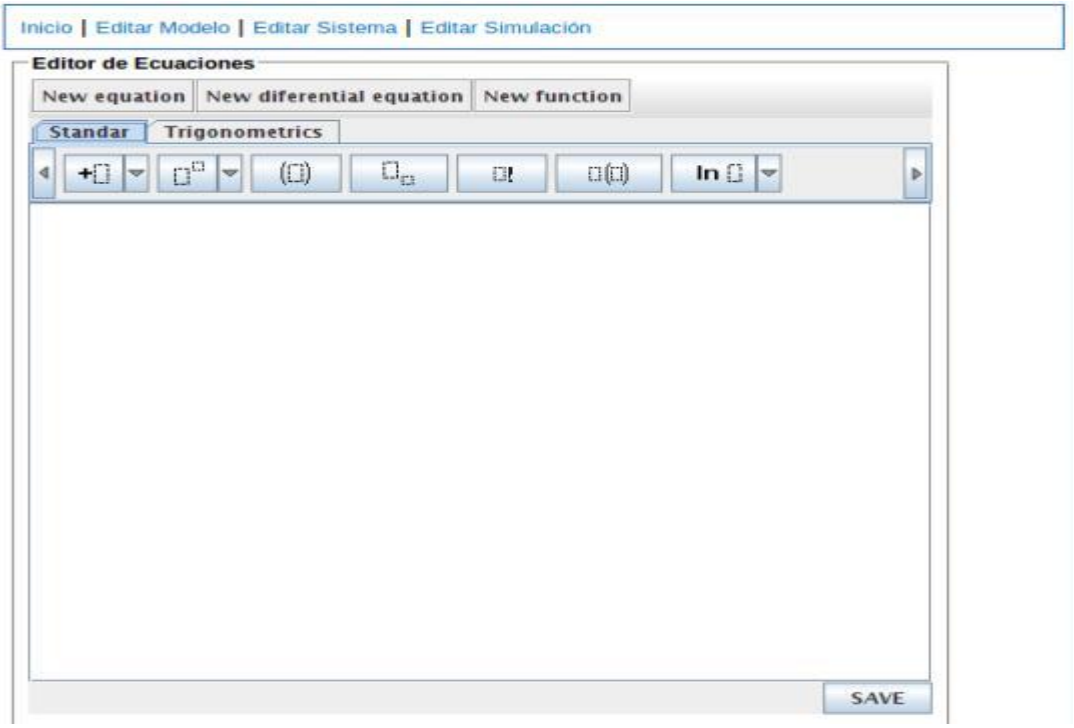
La imagen siguiente, representa la interfaz correspondiente a la configuración de sistemas biológicos.



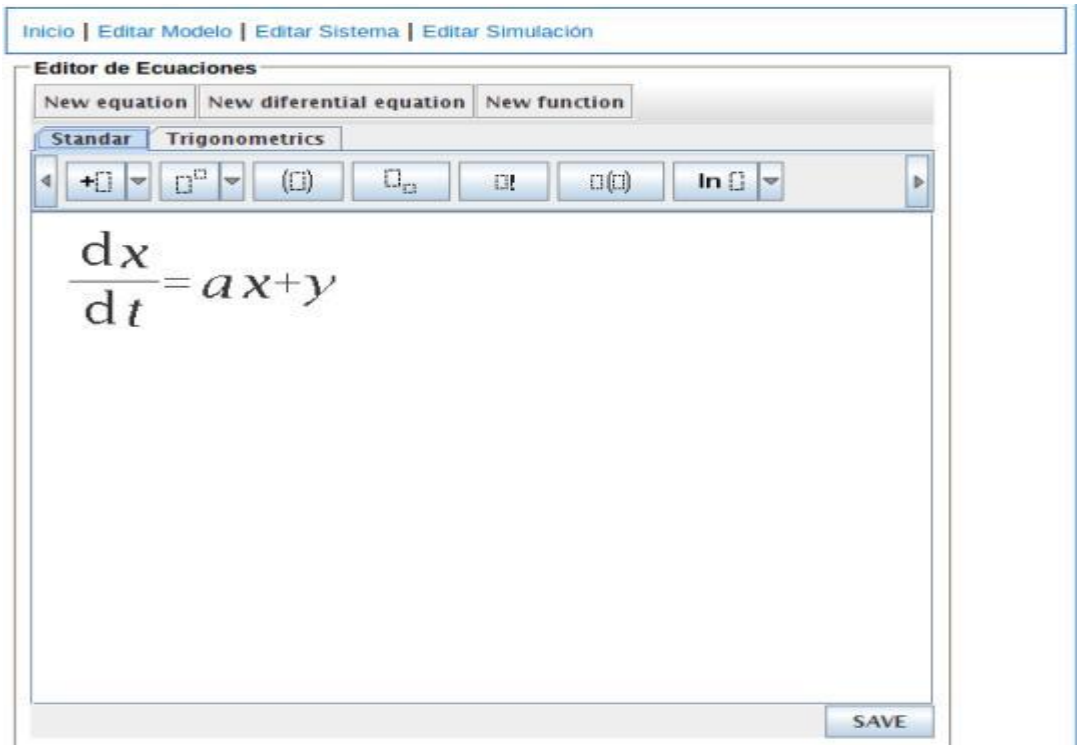
La imagen siguiente, representa la interfaz donde se muestran todos los sistemas biológicos registrados en la aplicación.

La imagen siguiente, representa la interfaz donde se muestran todos los modelos matemáticos del sistema biológico seleccionado, que se encuentren registrados en la aplicación.

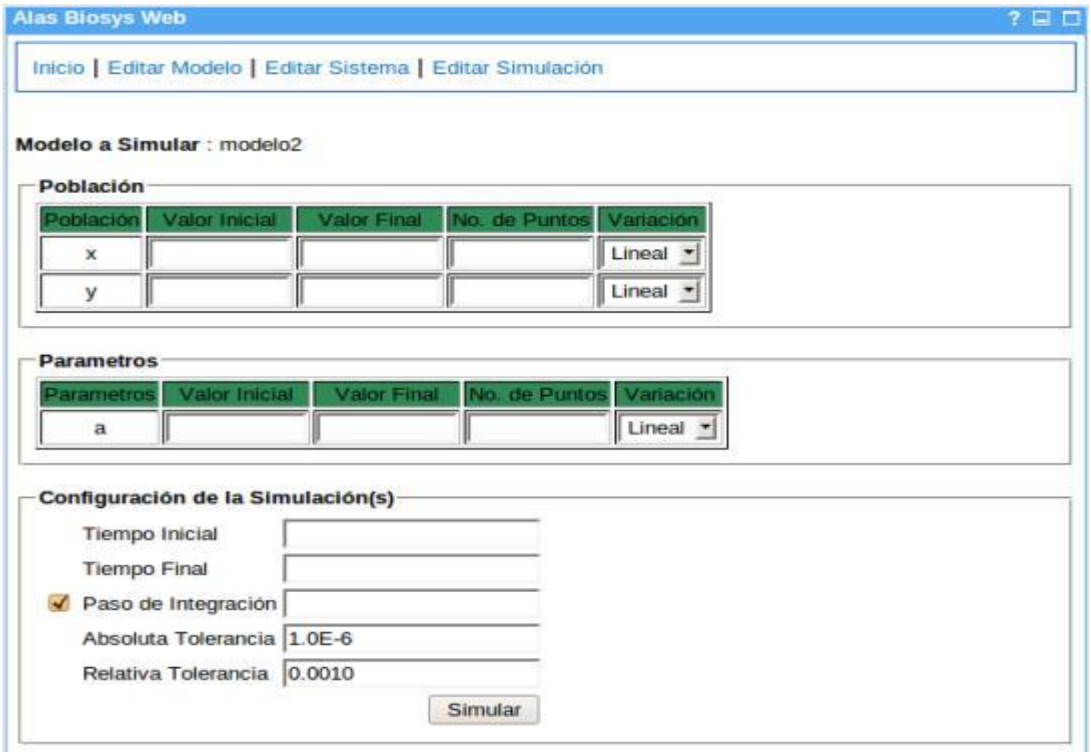
La imagen siguiente, representa la interfaz donde se muestra la herramienta Editor de Ecuaciones, donde el usuario puede insertar los datos para crear un nuevo modelo.



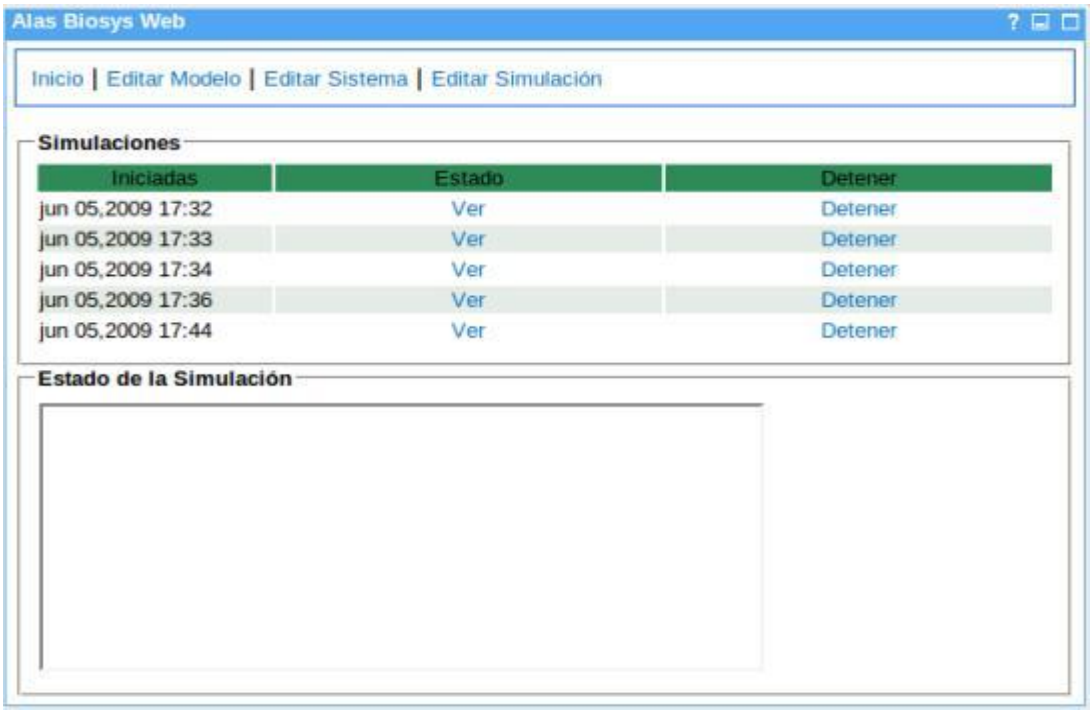
La imagen siguiente, representa la interfaz donde se muestra la herramienta Editor de Ecuaciones, donde el usuario puede modificar los datos del modelo seleccionado.



La imagen siguiente, representa la interfaz donde el usuario puede crear una nueva simulación.



La imagen siguiente, representa la interfaz correspondiente a la Gestión de la simulación.



4.5 Código Fuente de las principales clases

General

A continuación se muestra como se inserta un Sistema Biológico donde se utiliza la clase MainSystemPage.jsp y el método insertSystem de la clase Portlet_Biosys.java. Como resultado, debe adicionar el Sistema Biológico en el servidor.

En este código es donde el usuario inserta los datos del Sistema Biológico.

```
<ui:group key="NEWSYSTEM">
    <ui:form>
        <ui:table>
            <ui:tablerow>
                <ui:tablecell>
                    <ui:text key="NAME"></ui:text>
                </ui:tablecell>
                <ui:tablecell>
                    <ui:textfield maxlength="50" size="50"
beanId="name"></ui:textfield>
                </ui:tablecell>
            </ui:tablerow>
            <ui:tablerow>
                <ui:tablecell>
                    <ui:text key="DESCRIPTION"></ui:text>
                </ui:tablecell>
                 <ui:tablecell>
                     
                </ui:tablecell>
            </ui:tablerow>
            <ui:tablerow>
                <ui:tablecell colspan="2">
                     <ui:textarea rows="10" cols="58"
beanId="description"></ui:textarea>
                </ui:tablecell>
            </ui:tablerow>
//-----------------------------------------------------------1
            <ui:tablerow>
                <ui:tablecell>
                     
                </ui:tablecell>
                <ui:tablecell>
                    <ui:actionsubmit key="INSERT"
action="insertSystem"></ui:actionsubmit>

                </ui:tablecell>
            </ui:tablerow>
        </ui:table>
```

```
    </ui:form>
</ui:group>

//-------------------------------------------------------2
```

Explicación:

- 1 Se muestran los campos donde el usuario va a insertar el nombre y la descripción de Sistema Biológico.
- 2 Se especifica la acción que realizará el sistema una vez que el usuario presione el botón, donde se hace una llama al método insertSystem del Portlet.

Este método es llamado, cuando se presiona el botón INSERT del código de la clase anterior (MainSystemPage.jsp). Es el encargado de insertar el Sistema Biológico en el servidor, en caso de alguna dificultad se le envía un mensaje al usuario.

```
public void insertSystem(ActionFormEvent event) {
        String name = event.getTextFieldBean("name").getValue();
        String description = event.getTextAreaBean("description").getValue();
        MessageBoxBean msg = event.getMessageBoxBean("message");
        ActionRequest request = event.getActionRequest();
//--------------------------------------------------------------1
        try {
            if (name.trim().equals("")) {
                throw new Exception("empty name");
            }
            if (description.trim().equals("")) {
                throw new Exception("empty description");
            }
//--------------------------------------------------------------2
            Properties p = (Properties)
request.getPortletSession().getAttribute("bd");

            String host = p.getProperty("host").trim();
            String port = p.getProperty("port").trim();
            String database = p.getProperty("database").trim();
            String user = p.getProperty("user").trim();
            String pass = p.getProperty("password").trim();
            BSDAOFactory b = BSDAOFactory.getDAOFactory(BSDAOFactory.MYSQL, host,
port, database, user, pass);
//--------------------------------------------------------------3
```

```
            if (b.getSistemaDAO().getSistema(name) != null) {
                throw new Exception("duplicated system");
            }
//-------------------------------------------------------------------4
            b.getSistemaDAO().insertarSistema(new Sistema(name, description));
            msg.setMessageType(MessageStyle.MSG_SUCCESS);
            msg.setValue(getLocalizedText(request, "ADDED_SYSTEM"));
            List v = b.getSistemaDAO().getSistemas();
            b.Desconectar();
            request.getPortletSession().setAttribute("sistemas", v);
//-------------------------------------------------------------------5
        } catch (HibernateException ee) {
            msg.setMessageType(MessageStyle.MSG_ERROR);
            msg.setValue(getLocalizedText(request, "BD_UNAVAILABLE"));
        } catch (Exception e) {
            if (e.getMessage().equals("duplicated system")) {
                msg.setMessageType(MessageStyle.MSG_ERROR);
                msg.setValue(getLocalizedText(request, "DUPLICATED_SYSTEM_NAME"));
            } else if (e.getMessage().equals("empty name")) {
                msg.setMessageType(MessageStyle.MSG_INFO);
                msg.setValue(getLocalizedText(request, "EMPTY_SYSTEM_NAME"));
            } else if (e.getMessage().equals("empty description")) {
                msg.setMessageType(MessageStyle.MSG_INFO);
                msg.setValue(getLocalizedText(request, "EMPTY_SYSTEM_DESCRIPTION"));
            } else {
                msg.setMessageType(MessageStyle.MSG_ERROR);
                msg.setValue(e.getMessage());
            }

        }

        DEFAULT_VIEW_PAGE = "MainSystemPage.jsp";
        setNextState(request, "MainSystemPage.jsp");

    }
//-----------------------------------------------------------------------------------------------6
```

Explicación:

- 1 Se obtiene el nombre y la descripción del Sistema Biológico.
- 2 Se valida que ninguno de los campos, tanto el nombre como la descripción no estén vacíos.
- 3 Se establece la conexión a la Base de Datos.
- 4 Se valida que el nombre no haya sido registrado anteriormente en la Base de Datos.
- 5 Se inserta el Sistema Biológico en la Base de Datos en caso de no existir ningún problema y se le envía al usuario un mensaje informándole que el Sistema Biológico ha sido adicionado.
- 6 Se lanzan las excepciones correspondientes en caso de existir algún problema al insertar el

Sistema Biológico.

4.6 Pruebas de la aplicación

Prueba: es el proceso de ejecución de un programa con la intención de descubrir un error. [25]

En la aplicación se utilizó el método de Caja Negra, el cual estudia la especificación del software, se enfocan directamente en el exterior del módulo (pruebas a nivel de interfaz de usuario), sin importar el código. [25]

A continuación aparecen las pruebas realizadas a la aplicación.

Pruebas realizadas al CU_Gestionar Sistema.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo central
Insertar sistema biológico.	EC 1.1: Insertar sistema biológico correctamente.	Permite insertar el sistema biológico.	-Se presiona la opción Gestionar Sistema. -Se especifica el nombre del sistema biológico. -Se especifica la descripción del sistema biológico. -Se presiona el botón Insertar. -Se muestra un mensaje de confirmación.
	EC 1.2: Insertar sistema biológico dejando el campo del nombre en blanco.		-Se presiona la opción Gestionar Sistema. -Se especifica la descripción del sistema biológico. -Se presiona el botón Insertar. -Se muestra un mensaje de error.
	EC 1.3: Insertar sistema biológico dejando el campo de la descripción en blanco.		-Se presiona la opción Gestionar Sistema. -Se especifica el nombre del sistema biológico. -Se presiona el botón Insertar. -Se muestra un mensaje de error.
	EC 1.4: Insertar sistema biológico con el nombre existente.		-Se presiona la opción Gestionar Sistema. -Se especifica el nombre del sistema biológico. -Se especifica la descripción del sistema biológico. -Se presiona el botón Insertar.

	EC 1.5: Insertar sistema biológico sin tener conexión en la base de datos.		-Se presiona la opción Gestionar Sistema. -Se especifica el nombre del sistema biológico. -Se especifica la descripción del sistema biológico. -Se presiona el botón Insertar. -Se muestra un mensaje de error.
Modificar sistema biológico.	EC 2.1: Modificar sistema biológico correctamente.	Permite Modificar el sistema biológico.	-Se presiona la opción Configurar. -Se especifica el nombre del sistema biológico a modificar. -Se especifica la descripción del sistema biológico a modificar. -Se presiona el botón Modificar. -Se muestra un mensaje de confirmación.
	EC 2.2: Modificar sistema biológico dejando el campo del nombre en blanco.		-Se presiona la opción Configurar. -Se especifica la descripción del sistema biológico. -Se presiona el botón Modificar. -Se muestra un mensaje de error.
	EC 2.3: Modificar sistema biológico dejando el campo de la descripción en blanco.		-Se presiona la opción Configurar. -Se especifica el nombre del sistema biológico. -Se presiona el botón Modificar. -Se muestra un mensaje de error.
	EC 2.4: Modificar sistema biológico con el nombre existente.		-Se presiona la opción Configurar. -Se especifica el nombre del sistema biológico. -Se especifica la descripción del sistema biológico. -Se presiona el botón Modificar. -Se muestra un mensaje de error.
	EC 2.5: Modificar sistema biológico sin tener conexión en la base de datos.		-Se presiona la opción Configurar. -Se especifica el nombre del sistema biológico. -Se especifica la descripción del sistema biológico. -Se presiona el botón Modificar. -Se muestra un mensaje de error.
Eliminar sistema biológico.	EC 3.1: Eliminar sistema biológico correctamente	Permite Eliminar el sistema biológico.	-Se selecciona el sistema biológico. -Se presiona el botón Eliminar. -Se muestra un mensaje de confirmación.
	EC 3.2: Eliminar el sistema biológico sin tener conexión con la base de datos.		-Se selecciona el sistema biológico. -Se presiona el botón Eliminar. -Se muestra un mensaje de error.

Descripción de las variables

V1: Nombre(V- válido, I- inválido , NA- no necesario)

ID del escenario	Escenario	V1	Respuesta del sistema	Resultado de la prueba
EC 1.1	Insertar sistema biológico correctamente.	V	El sistema notifica que se insertó satisfactoriamente.	Satisfactorio
EC 1.2:	Insertar sistema biológico dejando el campo del nombre en blanco.	I	El sistema verifica que el campo insertado está incorrecto, muestra un mensaje de error informándole al usuario que el nombre es inválido.	Satisfactorio
EC 1.3:	Insertar sistema biológico dejando el campo de la descripción en blanco.	I	El sistema verifica que el campo insertado está incorrecto, muestra un mensaje de error informándole al usuario que la descripción es inválida.	Satisfactorio
EC 1.4:	Insertar sistema biológico con el nombre existente.	V	El sistema verifica y muestra un mensaje de error informando que el nombre ya existe en la base de datos.	Satisfactorio
EC 1.5:	Insertar sistema biológico sin tener conexión con la base de datos.	V	El sistema verifica y muestra un mensaje de error informando que no tiene acceso a la base de datos.	Satisfactorio
EC 2.1:	Modificar sistema biológico correctamente.	V	El sistema notifica que se modificó satisfactoriamente.	Satisfactorio
EC 2.2:	Modificar sistema biológico dejando el campo del nombre en blanco.	I	El sistema verifica que el campo insertado está incorrecto, muestra un mensaje de error informándole al usuario que el nombre es inválido.	Satisfactorio
EC 2.3:	Modificar sistema biológico dejando el campo de la descripción en blanco.	V	El sistema verifica que el campo insertado está incorrecto, muestra un mensaje de error informándole al usuario que el nombre es inválida.	Satisfactorio
EC 2.4:	Modificar sistema biológico con el nombre existente.	V	El sistema verifica y muestra un mensaje de error informando que el nombre ya existe en la base de datos.	Satisfactorio

EC 2.5:	Modificar sistema biológico sin tener conexión en la base de datos.	V	El sistema verifica y muestra un mensaje de error informando que no tiene acceso a la base de datos.	Satisfactorio
EC 3.1:	Eliminar sistema biológico correctamente	V	El sistema notifica que se eliminó satisfactoriamente.	Satisfactorio
EC 3.2:	Eliminar el sistema biológico sin tener conexión en la base de datos.	V	El sistema verifica y muestra un mensaje de error informando que no tiene acceso a la base de datos.	Satisfactorio

Pruebas realizadas al CU_Gestionar modelo matemático.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo central
Insertar Modelo.	EC 1.1: Insertar modelo correctamente.	Permite insertar un modelo.	-Se presiona la opción Insertar brindando la posibilidad de utilizar la herramienta editor de ecuaciones. -Se especifica el nombre del modelo y/o la ecuación diferencial. -Se presiona el botón Insertar. -Se guarda satisfactoriamente el modelo mostrando una notificación del mismo.
	EC 1.2: Insertar el modelo dejando el campo de nombre en blanco.		-Se presiona la opción Insertar brindando la posibilidad de utilizar la herramienta editor de ecuaciones. -Se presiona el botón Insertar. -Se muestra un mensaje de error.
	EC 1.3: Insertar el modelo sin tener conexión en la base de datos.		-Se presiona la opción Insertar. -Se especifica el nombre del modelo y/o la ecuación diferencial. -Se presiona el botón Insertar. -Se muestra un mensaje de error de que no exixte conexión alguna el la base de datos.
Modificar Modelo.	EC 2.1: Modificar modelo correctamente.	Permite Modificar el modelo deseado.	-Se presiona la opción Modificar. -Se verifica que no haya sido simulado el modelo, brindando la posibilidad de utilizar la herramienta editor de ecuaciones. -Se especifica la ecuación del modelo. -Se presiona el botón Modificar. -Se registra la información satisfactoriamente mostrando una notificación de la modificación del mismo.

	EC 2.2: Modificar modelo ya simulado o en proceso de simulación.		-Se presiona la opción Modificar. -Se verifica que el modelo ha sido simulado. -Se muestra un mensaje de error.
	EC 2.3: Modificar el modelo sin tener conexión en la base de datos.		-Se presiona la opción Modificar. -Se muestra un mensaje de error.
Eliminar sistema biológico.	EC 3.1: Eliminar el modelo correctamente	Permite Eliminar el modelo deseado.	-Se selecciona el modelo. -Se presiona el botón Eliminar. -Se muestra un mensaje de confirmación.
	EC 3.2: Eliminar el modelo sin tener conexión en la base de datos.		-Se presiona el botón Eliminar. -Se muestra un mensaje de error.
Realizar Simulación	EC 4.1: Simular el modelo correctamente	Permite realizar la simulación del modelo deseado.	-Se presiona la opción Simular del modelo seleccionado. -Se especifican los datos necesarios para realizar la simulación. -Se presiona el botón Simular. -Se guarda satisfactoriamente el modelo.
	EC 4.2: Simular el modelo con datos incorrectos.		-Se presiona la opción Simular del modelo seleccionado. -Se especifican los datos incorrectamente. -Se presiona el botón Simular. -Se muestra un mensaje de error.

Descripción de las variables

V1: Nombre(V- válido, I- inválido , NA- no necesario)

ID del escenario	Escenario	V1	Respuesta del sistema	Resultado de la prueba
EC 1.1	Insertar el modelo correctamente.	V	El sistema notifica que se insertó satisfactoriamente.	Satisfactorio
EC 1.2:	Insertar el modelo dejando el campo del nombre en blanco.	I	El sistema verifica que el campo insertado está incorrecto, muestra un mensaje de error informándole al investigador que el nombre es inválido.	Satisfactorio
EC 1.3:	Insertar el modelo sin tener conexión con la base de datos.	V	El sistema verifica y muestra un mensaje de error informando que no tiene acceso a la base de datos.	Satisfactorio

EC 2.1:	Modificar el modelo correctamente.	V	El sistema notifica que se modificó satisfactoriamente.	Satisfactorio
EC 2.2:	Modificar modelo ya simulado o en proceso de simulación.	I	El sistema verifica y muestra un mensaje de error informando que el modelo seleccionado ya ha sido simulado.	Satisfactorio
EC 2.3:	Modificar el modelo sin tener conexión con la base de datos.	V	El sistema verifica y muestra un mensaje de error informando que no tiene acceso a la base de datos.	Satisfactorio
EC 3.1:	Eliminar el modelo correctamente	V	El sistema notifica que se eliminó satisfactoriamente.	Satisfactorio
EC 3.2:	Eliminar el modelo sin tener conexión con la base de datos.	V	El sistema verifica y muestra un mensaje de error informando que no tiene acceso a la base de datos.	Satisfactorio
EC 4.1:	Simular el modelo correctamente	V	El sistema notifica que comenzó la simulación satisfactoriamente.	Satisfactorio
EC 4.2:	Simular el modelo con datos incorrectos.	I	El sistema verifica y muestra un mensaje de error informando que los datos introducidos son inválidos.	Satisfactorio

Pruebas realizadas al CU_Gestionar Simulación.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo central
Detener Simulación.	EC 1.1: Detener simulación correctamente.	Permite detener la simulación seleccionada.	-Se presiona la opción Gestionar Simulación. -Se presiona la opción Detener de la simulación deseada. -Se elimina la simulación de la base de datos. -Se muestra un mensaje de confirmación.
	EC 1.2: Detener simulación sin tener conexión con la base de datos.		-Se presiona la opción Gestionar Simulación. -Se presiona la opción Detener de la simulación deseada. -Se muestra un mensaje de error.

Ver Estado de la Simulación.	EC 2.1: Ver Estado de la simulación.	Permite ver el estado de la simulación seleccionada.	-Se presiona la opción Gestionar Simulación. -Se presiona la opción Ver Estado de la simulación deseada. -Se muestra el estado de la simulación.
	EC 2.2: Ver Estado de la simulación sin tener conexión con la base de datos.		-Se presiona la opción Gestionar Simulación. -Se presiona la opción Ver Estado de la simulación deseada. -Se muestra un mensaje de error.

Descripción de las variables

V1: Nombre(V- válido, I- inválido , NA- no necesario)

ID del escenario	Escenario	V1	Respuesta del sistema	Resultado de la prueba
EC 1.1	Detener simulación correctamente.	V	El sistema notifica que la simulación fue detenida satisfactoriamente.	Satisfactorio
EC 1.2:	Detener simulación sin tener conexión con la base de datos.	V	El sistema verifica y muestra un mensaje de error informando que no tiene acceso a la base de datos.	Satisfactorio
EC 2.1:	Ver Estado de la simulación.	V	El sistema muestra el estado de la simulación.	Satisfactorio
EC 2.2:	Ver Estado de la simulación sin tener conexión con la base de datos.	V	El sistema verifica y muestra un mensaje de error informando que no tiene acceso a la base de datos.	Satisfactorio

Conclusiones

En este capítulo se logra el resultado del diseño, implementando el sistema en términos de componentes, realizando el diagrama de componentes. También se presenta el código fuente de las principales clases y pantallas de la aplicación como por ejemplo prototipos de interfaz de usuario y el mapa de Navegación, además se exponen los resultados de las pruebas de aceptación desarrolladas utilizando el método de caja negra.

CONCLUSIONES

Todos los objetivos específicos que se trazaron en el trabajo de diploma fueron cumplidos satisfactoriamente:

- Se definieron correctamente todas las funcionalidades de la aplicación alas BioSyS Web.
- Se diseñó satisfactoriamente la arquitectura candidata de la aplicación Web para gestionar las funcionalidades seleccionadas de alas BioSyS.
- Se implementó la aplicación Web correspondiente para gestionar las funcionalidades seleccionadas de alas BioSyS.

RECOMENDACIONES

Esta aplicación puede ampliar sus perspectivas, por lo que se recomienda:

- Ampliar la aplicación de manera que cumpla con todas las funcionalidades del software alas BioSyS.
- Actualizarlo acorde a las versiones correspondientes a la aplicación de escritorio.

REFERENCIAS BIBLIOGRAFICAS

[1]. Teoría General de Sistemas. [Online] teoriageneraldesistemasunefa.blogspot.com/2009/05/exposiciones-seccion-5n4is_19.html.

[2]. Departamento de Biología de Sistemas. [Online] http://www.uvic.cat/eps/dept/biologiasistemes/es/inici.html.

[3]. **Durán, Niurka Martínez.** *Portal Web de Servicios Bioinformaticos.* la Habana : s.n., 2008.

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

[5]. **Tadmor B, Tidor B.** *Interdisciplinary research and education at the biology engineering computer science interface: a perspective .* 2005. 17.

[6]. **Ehrenberg M, Elf J, Aurell E, Sandberg R, Tegner J.** *Systems Biology Is Taking Off. Genome Res.* 2003. 13:2377–2380.

[7]. **Lemus, Noel Moreno.** *BioSyS: Software para la simulación y análisis de sistemas biológicos.* la Habana : s.n., 2007.

[8]. **John Brewer, Jera Design.** Extreme Programming FAQ. [Online] http://www.jera.com/techinfo/xpfaq.html.

[9]. **Requena, Martín Luis López.** Microsoft Solutions Framework. [Online] http://www.malagadnug.org/ficheros/MSFMartinLuisReq.pdf.

[10]. **Torossi, Gustavo.** El Proceso Unificado de Desarrollo de Software. [Online] http://www.chaco.gov.ar/UTN/disenodesistemas/apuntes/oo/ApunteRUP.pdf.

[11]. Open UP. [Online] http://epf.eclipse.org/wikis/openup/.

[12]. Visual Paradigm IDE for Java. [Online] http://www.visual-paradigm.com/.

[13]. **Grady Booch, James Rumbaugh E Ivar Jacobson.** *El Lenguaje Unificado De Modelado, Guía Del Usuario.*

[14]. Java en castellano. [Online] http://www.programacion.com/java/.

[15]. **José Zerpa, Javier Escobar y Yonel Meza.** Desarrollo de Aplicaciones Basadas en Ambientes Virtualizados Bajo Software Libre y Estándares Abiertos, a Través de la Metodología de Producción Acelerada SCRUM. [Online] http://www.onuva.com/files/OnuvaConfl/OnuvaConflDesarrollo.pdf.

[16]. Entornos de Desarrollo Integrado para Java. [Online] http://luauf.com/2008/05/13/entornos-de-desarrollo-integrado-para-java/.

[17]. **Dapena, Martha D. Delgado.** Definición del modelo del negocio y del dominio utilizando Razonamiento Basado en Casos. [Online] http://www.inf.udec.cl/~revista/ediciones/edicion8/Rbc.pdf.

[18]. **González, Guillermo.** Ingeniería de Requerimientos. [Online] 2008. http://trevinca.ei.uvigo.es/~ebalonso/asignaturas/esx/guiones/esxClase6.pdf.

[19]. Ingeniería de Software 1. Flujo de Trabajo de Requerimientos. [Online] 2008.

[20]. **Astudillo, Marcello Visconti y Hernán.** Fundamentos de Ingeniería de Software. [Online] http://www.inf.utfsm.cl/~visconti/ili236/Documentos/10-DisenoOO.pdf.

[21]. **Lien Le Sánchez, Reynaldo Rosado Roselló. 2008.** *alasARBOGEN: aplicación informática para la representación de árboles genealógicos.* la Habana : s.n., 2008.

[22]. Arquitectura y Patrones de diseño. Ingenieria de Software. [Online] 2008.

[23]. Ingeniería del Software (3º I.T.I.S., I.T.I.G.) Módulo 2. [Online] http://www.dsi.uclm.es/asignaturas/42530/pdf/M2tema12.pdf.

[24]. **Cano, Sandra P.** MODELAMIENTO DE APLICACIONES WEB. [Online] http://www.netwebsys.net/blog/files/aplic_web.pdf.

[25]. **Quesada., Juan Antonio López.** Ingeniería del Software. [Online] http://dis.um.es/~lopezquesada.

BIBLIOGRAFIA

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

2. **Adomian G, Adomian GE, Bellaman RE. 1984.** *Biological System Interactions. Physiological. Sciences.* 1984. 81:2838–2940.

3. Almendras Vargas Wilfredo Ajayu[En línea] septiembre de 2007. [Online] http://ajayu.memi.umss.edu.bo/wilfredo/weblog/patrones-de-disenno.

4. Arquitectura y Patrones de diseño. Ingenieria de Software. [Online] 2008.

5. **Astudillo, Marcello Visconti y Hernán.** Fundamentos de Ingeniería de Software. [Online] http://www.inf.utfsm.cl/~visconti/ili236/Documentos/10-DisenoOO.pdf.

6. **Cano, Sandra P.** MODELAMIENTO DE APLICACIONES WEB. [Online] http://www.netwebsys.net/blog/files/aplic_web.pdf.

7. **Dapena, Martha D. Delgado.** Definición del modelo del negocio y del dominio utilizando Razonamiento Basado en Casos. [Online] http://www.inf.udec.cl/~revista/ediciones/edicion8/Rbc.pdf.

8. **Durán, Niurka Martínez. 2008.** *Portal Web de Servicios Bioinformaticos.* la Habana : s.n., 2008.

9. **Ehrenberg M, Elf J, Aurell E, Sandberg R, Tegner J. 2003.** *Systems Biology Is Taking Off. Genome Res.* 2003. 13:2377–2380.

10. **González, Guillermo. 2008.** Ingeniería de Requerimientos. [Online] 2008. http://trevinca.ei.uvigo.es/~ebalonso/asignaturas/esx/guiones/esxClase6.pdf.

11. Ingeniería de Software 1. Flujo de Trabajo de Requerimientos. [Online] 2008.

12. Ingeniería del Software (3º I.T.I.S., I.T.I.G.) Módulo 2. [Online] http://www.dsi.uclm.es/asignaturas/42530/pdf/M2tema12.pdf.

13. **Josep Bau Macia, Joan Bertran Comulada.** Departamento de Biología de Sistemas. [Online] http://www.uvic.cat/eps/dept/biologiasistemes/es/inici.html.

14. **Lemus, Noel Moreno. 2007.** *BioSyS: Software para la simulación y análisis de sistemas biológicos.* la Habana : s.n., 2007.

15. **Letelier P, Penades C. 2002.** *Metodologías ágiles para el desarrollo de software: eXtreme Programming.* 2002.

16. **Lien Le Sánchez, Reynaldo Rosado Roselló. 2008.** *alasARBOGEN:aplicación informática para la representación de árboles genealógicos.* La Habana : s.n., 2008.

17. **Luscombe NM, Greenbaum D, Gerstein M. 2001.** *What is bioinformatics? An introduction and overview.* s.l. : Yearbook of Medical Informatics, 2001.

18. **Quesada., Juan Antonio López.** Ingeniería del Software. [Online] http://dis.um.es/~lopezquesada.

19. **Tadmor B, Tidor B. 2005.** *Interdisciplinary research and education at the biology engineering computer science interface: a perspective .* 2005. 17.

20. Teoría General de Sistemas. [Online] teoriageneraldesistemasunefa.blogspot.com/2009/05/exposiciones-seccion-5n4is_19.html.

21. Entornos de Desarrollo Integrado para Java. [Online] http://luauf.com/2008/05/13/entornos-de-desarrollo-integrado-para-java/.

22. **Grady Booch, James Rumbaugh E Ivar Jacobson.** *El Lenguaje Unificado De Modelado, Guía Del Usuario.*

23. Java en castellano. [Online] http://www.programacion.com/java/.

24. **John Brewer, Jera Design.** Extreme Programming FAQ. [Online] http://www.jera.com/techinfo/xpfaq.html.

25. **José Zerpa, Javier Escobar y Yonel Meza.** Desarrollo de Aplicaciones Basadas en Ambientes Virtualizados Bajo Software Libre y Estándares Abiertos, a Través de la Metodología de Producción Acelerada SCRUM. [Online] http://www.onuva.com/files/OnuvaConfI/OnuvaConfIDesarrollo.pdf.

26. Open UP. [Online] http://epf.eclipse.org/wikis/openup/.

27. **Requena, Martín Luis López.** Microsoft Solutions Framework. [Online] http://www.malagadnug.org/ficheros/MSFMartinLuisReq.pdf.

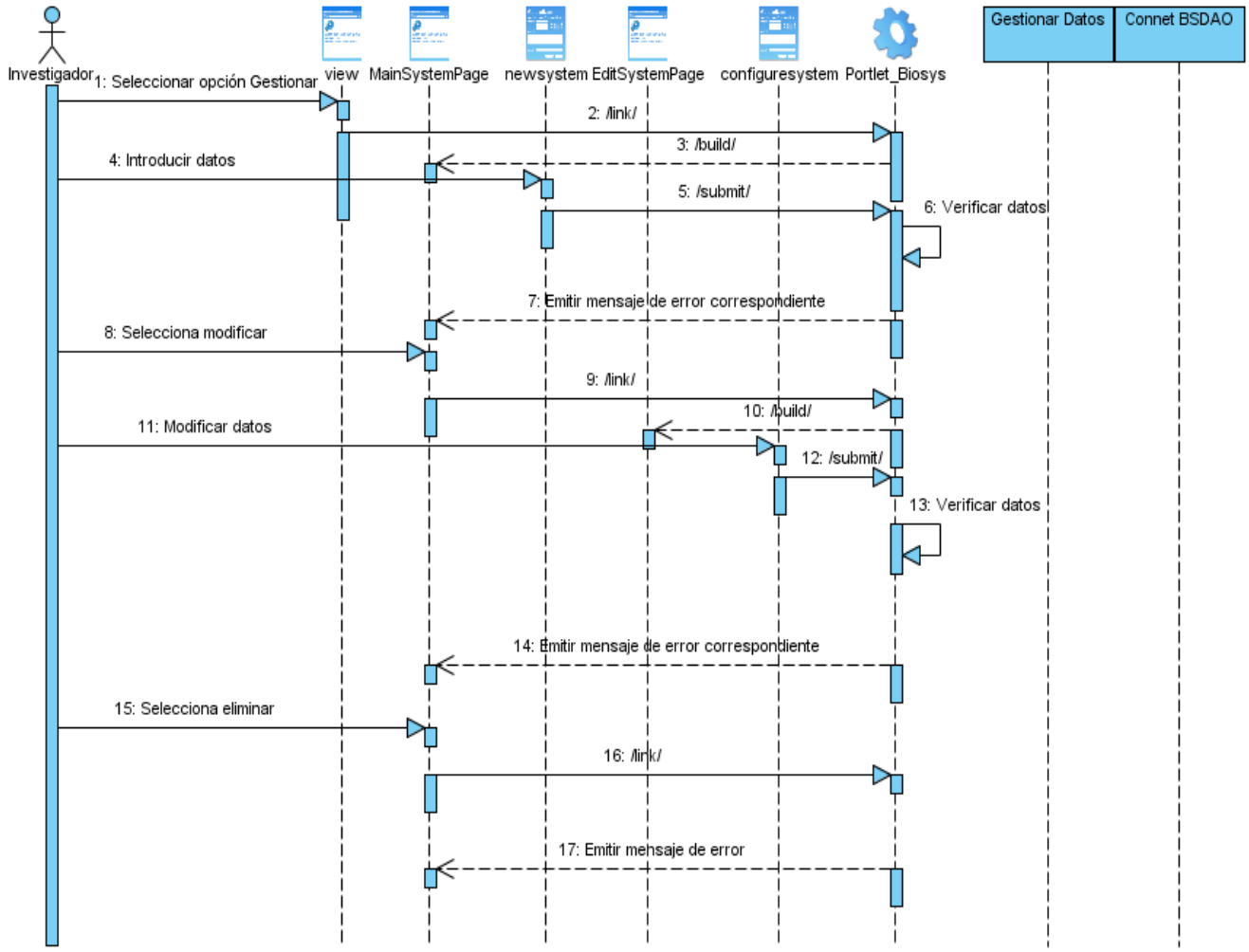
28. **T, Keane. 2003.** Java Distributed System: Developer Manual. [Online] 2003. http://www.cs.may.ie/.

29. **Torossi, Gustavo.** El Proceso Unificado de Desarrollo de Software. [Online] http://www.chaco.gov.ar/UTN/disenodesistemas/apuntes/oo/ApunteRUP.pdf.

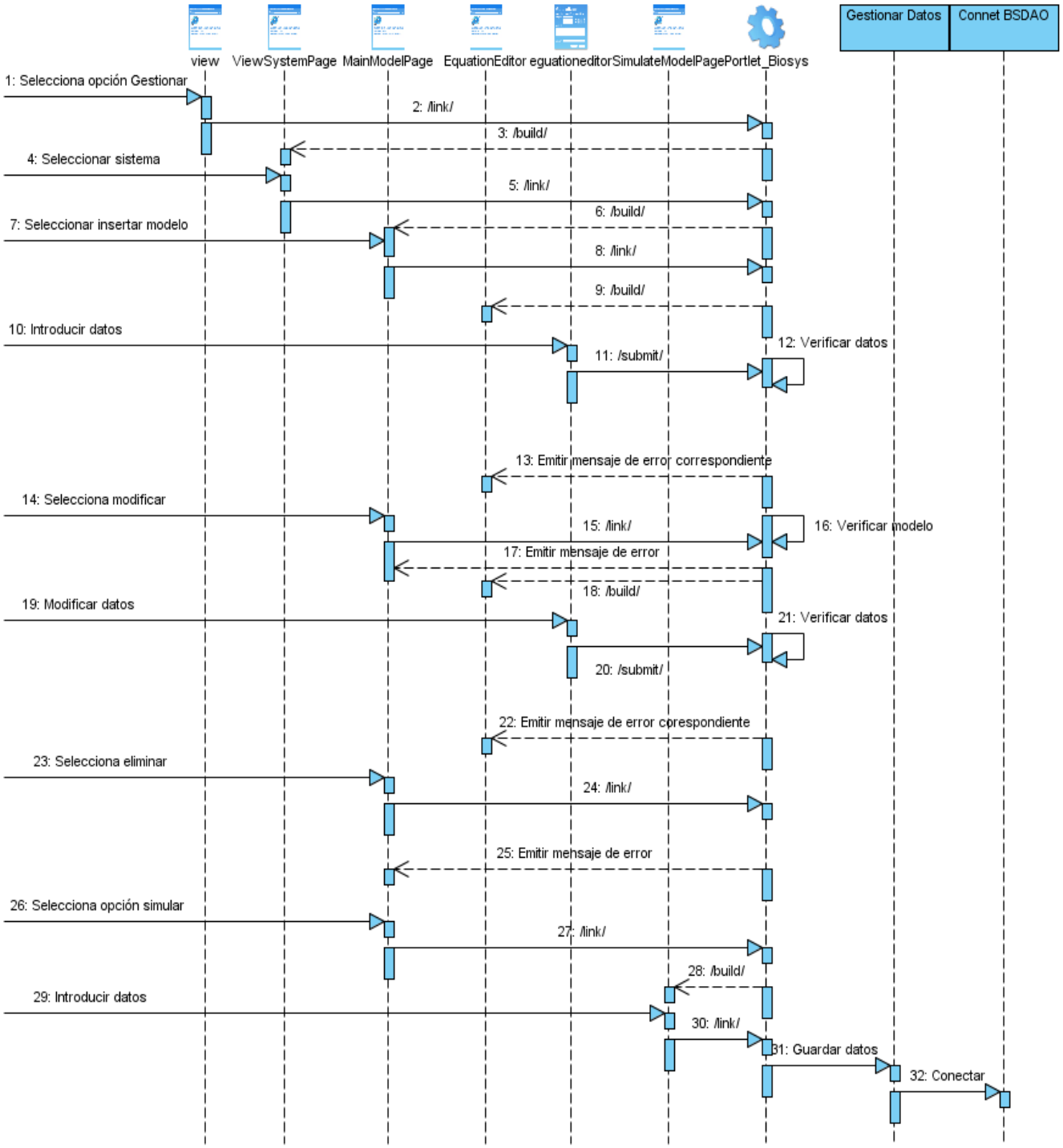
30. Visual Paradigm IDE for Java. [Online] http://www.visual-paradigm.com/.

ANEXOS

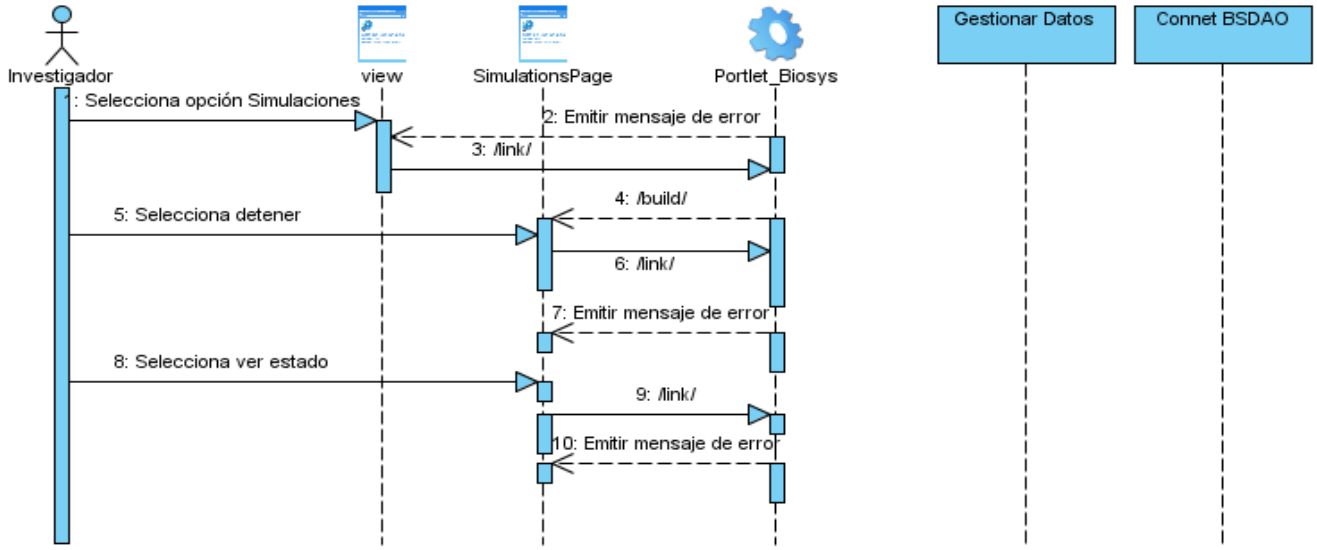
A continuación se muestran los Diagramas de Secuencia del flujo alterno correspondientes a los Casos de Uso Gestionar sistema biológico, Gestionar modelo matemático y Gestionar simulación, ver anexo 1, anexo 2 y anexo 3 respectivamente.



Anexo 1 Diagrama de Secuencia del Flujo Alterno del CU_Gestionar sistema biológico



Anexo 2 Diagrama de Secuencia del Flujo Alterno del CU_Gestionar modelo matemático



Anexo 3 Diagrama de Secuencia del Flujo Alterno del CU_Gestionar simulación

GLOSARIO

(CIGB)	Centro de Ingeniería Genética y Biotecnología
(CIM)	Centro de Inmunología Molecular
(alas BioSyS)	Biological System Simulator
(API)	Applications Programming Interface
(SOAP)	Simple Object Access Protocol
(UDDI)	Universal Description Discovery and Integration
(WSDL)	Web Service Description Language
(XP)	Extreme Programing o Programación extrema
(MSF)	Microsoft Solution Framework
(RUP)	Proceso unificado de desarrollo de software
(CASE)	Computer Aided Software Engineering
(GNU/LINUX)	Sistema operativo Ubuntu
(ODL)	Object Definition Lenguage
(UML)	Unified Modeling Languaje
(CVS)	Concurrent Versions System
(IDE)	Integrated Development Environment

(MVC)	Modelo Vista Controlador
(CRU)D	Creating, Reading, Updating, Deleting