Universidad de las Ciencias Informáticas

Facultad 6

Título: "Sistema para el Análisis de Sensibilidad en la plataforma BioSyS"

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

Autor: Jorge Luis Esperanza Fernández

Tutor(es): Msc. Yudelkis Abad Fuentes

Msc. Yunet González Mulet

2011, La Habana, Cuba

“Cuando se es 'hombre de ciencia', no se tiene ideal: se elaboran resultados científicos, y cuando se es hombre de partido se combate para ponerlos en práctica. Pero cuando se tiene un ideal, no se puede ser hombre de ciencia, pues se ha adoptado una decisión de antemano”.

Ernesto Guevara de la Serna

Declaración de autoría

Declaro ser el único autor del presente trabajo y autorizo a la Facultad 6 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Jorge Luis Esperanza Fernández

_____

Firma del Autor

Msc. Yudelkis Abad Fuentes

_____

Firma del Tutor

Msc. Yunet González Mulet

_____

Firma del Tutor

Datos de contacto

Tutor: Msc. Yudelkis Abad Fuentes

Especialidad de graduación: Máster en bioinformática

Categoría docente: Instructor

Categoría científica: Máster

Años de experiencia en el tema: 4

Años de graduado: 4

Correo electrónico: yabad@uci.cu

Tutor: Msc. Yunet González Mulet

Especialidad de graduación: Máster en bioinformática

Categoría docente: Instructor

Categoría científica: Máster

Años de experiencia en el tema: 7

Años de graduado: 4

Correo electrónico: ygonzalezmu@uci.cu

Agradecimientos

A aquellos que han colaborado con mi formación como profesional, a mis padres, familiares y amigos que me han apoyado a lo largo de la carrera, a mi novia por su cariño, comprensión y ayuda en todo momento. A la revolución y a Fidel por haberme dado la posibilidad de estudiar en una universidad de exelesia, a mis profesores por formarme como ingeniero en estos cinco año, y en general a todos los que estuvieron a mi lado compartiendo las buenas y las malas. Gracias.

Dedicatoria

- *A mis padres: Sandra e Idalberto, por tener confianza en mí, y por haberme dado fuerzas para terminar lo que hace 5 años comencé y hoy con mucho sacrificio termino.*
- *A mi padrastro Ruber, por la buena educación que me ha dado, por todo su cariño y comprensión en los buenos y malos momentos.*
- *A mi tía Tamara que siempre ha estado en los buenos y en los malos momentos cuando la he necesitado.*
- *A mis abuelos Otalina, Caridad, Manuel y Alfonso, que los quiero mucho y aunque algunos no pueden estar aquí porque están muertos se que estarían muy orgullosos en este momento.*
- *A mis hermanos y primos.*
- *A toda mi familia y seres queridos.*
- *A mis tutoras Yunet y Yudelkis, por todo su apoyo y ayuda.*
- *A Edel, por su incondicional apoyo en todos los momentos en que lo necesité.*
- *A mis amigos de la Universidad, en especial a Yasniel, Héctor Alcides, Hector Yasmani, Leonel, Elio, Flavio y muchos más a los cuales no menciono, porque creo que no terminaría nunca. Que ha tenido que soportarme todo este tiempo.*
- *A todos mis amigos de Las Tunas, Yuslier, Falvio, Carlos, Ariel, Alexis y Yoel.*
- *A mi novia Yudith, por su comprensión, paciencia, apoyo y cariño en todo momento.*
- *A mis compañeros y hermanos Isael y Yunier que me han demostrado que existe la verdadera amistad.*

Resumen

El Análisis de Sensibilidad en modelos matemáticos es un tema muy reciente en Biología de Sistemas. Esta investigación examina los principales métodos de Análisis de Sensibilidad en modelos matemáticos. Actualmente un conjunto de especialistas en Biología de Sistemas del Centro de Inmunología Molecular (CIM) y la Universidad de las Ciencias Informáticas (UCI), ambos de Cuba, han desarrollado una aplicación llamada BioSyS, con el objetivo de facilitar el proceso de modelado y simulación de sistemas biológicos en el campo de la Biología de Sistemas, pero que aún carece de este tipo de análisis. Este trabajo realiza un estudio de los principales conceptos y métodos relacionados con Análisis de Sensibilidad en modelos matemáticos de sistemas biológicos, se analizan los softwares existentes y propone una solución modular a la problemática, de modo que se pueda integrar con la plataforma BioSyS.

Palabras Claves: Análisis de Sensibilidad, modelos matemáticos, Biología de Sistemas, BioSyS.

Tabla de contenido

Índice de tablas

Índice de figuras

Introducción

Cada vez existe un mayor acercamiento entre la tecnología y los procesos sociales. En este ámbito de transformaciones se encuentran las ciencias biológicas e informáticas, proceso que ha dado lugar en las condiciones actuales al término: Bioinformática, marco en donde se desarrolla la presente investigación y cuyo objetivo está asociado, a los procesos de Análisis de Sensibilidad en sistemas de ecuaciones diferenciales, técnica que hoy día, es imprescindible en el análisis y simulación de sistemas biológicos.

El Análisis de Sensibilidad es una técnica que se usa para conocer cuánto influye la variabilidad de valores de un parámetro desconocido en la salida del sistema. Como resultado de este proceso, se sabrá si dicho parámetro es sensible o no. Esta técnica es muy utilizada para evaluar el impacto que tienen los datos de entrada o restricciones especificadas a un modelo definido, en el resultado final o en las variables de salida de dicho modelo. (1)

Para ello se debe partir de la resolución del modelo matemático que describe el comportamiento del sistema. En esta modelación matemática del fenómeno, existen parámetros conocidos y parámetros que el usuario debe estimar –a partir de técnicas existentes– debido a su desconocimiento. Si la cantidad de parámetros a estimar, y el posible rango definido para cada uno es amplio, el proceso de estimación se vuelve lento, por lo que se requiere de la técnica de análisis de sensibilidad para destacar los parámetros que realmente inciden en la salida del sistema y decidir sobre cuáles variables debe enfocarse el investigador.

Por la importancia antes comentada, se han creado varias aplicaciones que se implementan en dicho análisis, entre las que se encuentran: COPASI, XPPAUT, EASY-FIT EXPRESS 1.0, BERKELEY MADONNA v8.0.1, esta última es una de las más usadas en la actualidad para resolver ecuaciones diferenciales. Pero son herramientas en su mayoría sin acceso a código fuente, propietarias, sin derecho a modificación y no comerciales, deficiencias estas, que limitan el trabajo de los investigadores en la rama.

En Cuba existen centros que se dedican a darle continuidad a estos estudios, como son: el Centro de Ingeniería Genética y Biotecnología (CIGB) y el Centro de Inmunología Molecular (CIM). La Universidad de las Ciencias Informáticas en conjunto con este último, ha desarrollado un software llamado BioSyS -del inglés, Biological Systems Simulator / Simulador de Sistemas

Biológicos- que realiza exploraciones intensivas en sistemas biológicos, descritos mediante sistemas de ecuaciones diferenciales, y permite la estimación de los parámetros deseados, pero no cuenta con una herramienta propia para realizar Análisis de Sensibilidad de los parámetros que describen al modelo matemático, para así acotar la búsqueda de las estimaciones de parámetros realizadas.

Este software se encuentra organizado por módulos, donde cada uno se encarga de funcionalidades distintas en temáticas específicas, por ejemplo: el Módulo Editor de Ecuaciones se encarga de la modelación matemática del sistema; el Módulo Estimación se encarga de calcular los valores de las variables y parámetros desconocidos; el Módulo Simulación se encarga de resolver dichas ecuaciones y el de Análisis se encarga de estudiar los resultados obtenidos. Sin embargo, BioSyS no realiza el Análisis de Sensibilidad de las variables y parámetros desconocidos para así disminuir el tiempo de cómputo de la estimación.

Por lo que surge como **Problema a resolver:** Inexistencia de un módulo en BioSyS que realice análisis de sensibilidad.

El **objeto de estudio** de la presente investigación es: El proceso de Análisis de Sensibilidad. A partir del objeto de estudio se delimita, como **campo de acción:** Análisis de Sensibilidad en sistemas de ecuaciones diferenciales.

Para solucionar esta problemática se ha propuesto como **objetivo general:** Desarrollar un Sistema para el Análisis de Sensibilidad en la plataforma BioSyS.

Para cumplir este objetivo general se plantearon los siguientes **objetivos específicos**:

- Realizar estudio del arte sobre la técnica de Análisis de Sensibilidad.
- Realizar el diseño de la aplicación para el Análisis de Sensibilidad en la plataforma BioSyS.
- Implementar la aplicación diseñada para el Análisis de Sensibilidad en la plataforma BioSyS.
- Validar la aplicación implementada para el Análisis de Sensibilidad en la plataforma BioSyS.

Para alcanzar dichos objetivos específicos se plantearon las siguientes tareas:

- Análisis del estado de conocimiento sobre la técnica de Análisis de Sensibilidad.
- Identificación de los requisitos funcionales y no funcionales de la aplicación.
- Identificación de los Casos de Uso del Sistema.
- Construcción del Diagrama de Casos de Uso del Sistema.
- Especificación de los Casos de Uso del sistema.
- Construcción de los Diagrama de clases del diseño.
- Definición del algoritmo para el método de Análisis de Sensibilidad escogido.
- Implementación de los Casos de Uso del Sistema.
- Aplicación de prueba de caja negra a la aplicación.

Este trabajo ha sido organizado en cuatro capítulos de la siguiente manera:

Capítulo I: En este capítulo se realiza un estudio del arte acerca del Análisis de Sensibilidad. Se exponen las tendencias, técnicas, tecnologías, metodología y los lenguajes de programación seleccionados para darle solución al problema planteado.

Capítulo II: En este capítulo se realiza el diseño de la aplicación basada en la metodología OpenUp, la cual establece primeramente la definición de los actores del sistema, posteriormente los requisitos funcionales para la modelación del Diagrama de Casos de Usos del Sistema y, los requisitos no funcionales. Finalmente propone la modelación del Diagrama de Clases del Diseño y el Diagrama de Interacción por cada caso de uso.

Capítulo III: En este capítulo se diseñó el Diagramas Despliegue y el Diagrama de Componentes, en el cuál se representan los principales componentes y sus relaciones. También se realizó la implementación del sistema. Para ello se muestra el código fuente de los principales métodos y se realizan las pruebas necesarias para validar la solución.

Capítulo 1: Fundamentación teórica

1. Introducción

En este capítulo se realiza un estudio del arte acerca del Análisis de Sensibilidad. Se exponen las tendencias, técnicas, tecnologías, metodología y los lenguajes de programación seleccionados para darle solución al problema planteado.

1.1 Análisis de Sensibilidad

El Análisis de Sensibilidad es una técnica que se usa para conocer cuán perceptible es el sistema en estudio, a los cambios en los valores de los parámetros desconocidos, y el resultado final es la aceptación o no de la sensibilidad del parámetro. Esta técnica es muy utilizada para evaluar el impacto que tienen los datos de entrada o restricciones especificadas a un modelo definido, en las variables de salida de dicho modelo. (1)

Generalmente su uso está dado en:

1. Simplificar modelos.
2. Explorar el impacto de la variación de entradas y escenarios.
3. Identificar las variables más críticas.
4. Investigar las fortalezas de las predicciones de un modelo.

En este análisis generalmente se tienen en cuenta los siguientes pasos:

Paso (1) Seleccionar los parámetros a ser analizados: es donde el usuario determina los valores para cada parámetro que interviene en el modelo, describiendo las características del mismo y especificando si va a ser fijo o variable. En el caso que sea variable, será el parámetro a estimar.

Paso (2) Fijar el rango de valores para cada parámetro seleccionado. Se define el rango en que va a estar el valor del parámetro. Este rango debe ser lo suficientemente grande para poder cubrir todas las variaciones posibles.

Paso (3) Generar una serie de números con una distribución uniforme dentro del rango. En este paso, el usuario define en cuántos puntos dentro del rango definido en el paso (2), se va a resolver el modelo y evaluar la función.

Paso (4) Resolver el modelo para cada set de valores y calcular la función objetivo correspondiente. Los valores obtenidos son graficados en correspondencia con la función objetivo.

Paso (5) Determinar si el set de valores escogido para ese parámetro es aceptable o inaceptable, comparando el valor resultante calculado para cada set del parámetro de entrada, con el promedio de los valores calculados por la función objetivo (criterio de comparación). Si el valor calculado es mayor que el criterio, el juego de valores del parámetro es clasificado como inaceptable; por el contrario, es clasificado como aceptable.

Paso (6) Se evalúa la sensibilidad del parámetro comparando dos distribuciones de valores del parámetro asociadas a los resultados de "aceptables e inaceptables". Si las dos distribuciones no son similares (coeficiente de correlación relativamente grande), el parámetro es clasificado como sensible; de lo contrario, el parámetro es clasificado como insensible.

En esencia, la mayoría de los trabajos reportados en la literatura se rigen por los pasos descritos anteriormente, lo que varían son las funciones objetivo definidas, y la evaluación de la sensibilidad. (2) (3) (4) (5) (6)

Para darle solución a lo antes expuesto, en los pasos (3) y (4) se debe utilizar un método matemático que permita generar valores aleatorios dentro del rango definido para cada una de las variables de entrada que fija el investigador y resolver el modelo matemático. Luego el sistema comienza a resolver la función de transferencia (función objetivo) para cada uno de los valores aleatorios dentro del rango, una vez terminado este proceso, calcula la media de todas las soluciones obtenidas.

1.2 Técnicas para realizar Análisis de Sensibilidad

Al considerar las variaciones de los parámetros, en la evaluación de un modelo, se puede actuar de dos maneras:

- Variar un parámetro manteniendo constante los restantes.
- Variaciones combinadas de todos ellos.

El procedimiento más completo es indudablemente este último; aunque presenta el inconveniente que al poderse realizar multitud de posibles combinaciones, exige la utilización de una computadora para su resolución.

Sin embargo, para modelos más simples, se puede considerar el efecto individual de las variables más importantes. Asimismo, desde un punto de vista operativo, el primer método puede servir para identificar las variables críticas o más influyentes y una vez determinadas se puede pasar a realizar variaciones combinadas de estas técnicas.

En la presente investigación se utiliza la técnica de variaciones combinadas.

1.3 Métodos matemáticos para el Análisis de Sensibilidad

- FAST (Fourier Amplitude Sensitivity Test / Prueba de Sensibilidad de Amplitud de Fourier) es uno de los procedimientos implementados para estimar la variación de la salida de un parámetro en estudio. Este método es independiente de cualquier hipótesis sobre la estructura del modelo (Saltelli et al., 2000). Es generalmente usado en modelos sin interacciones importantes o significativas entre los parámetros (Saltelli y Bolado, 1998).

El método FAST no es eficiente utilizarlo en interacciones de un gran número de entradas (Saltelli& Bolado, 1998). Además, la fiabilidad puede ser pobre para entradas discretas (Saltelli et al., 2000).

- RSM (Response Surface Method / Método de Respuesta a Superficie) es usado mayormente en análisis probabilísticos (Frey & Bharvirkar, 1998; Chun et al., 1996). Su objetivo principal es desarrollar una versión simplificada del modelo original para que sea posible retener las características claves del modelo y acortar el de tiempo requerido para

predecir la salida de un determinado juego de entradas. En general es complejo y por lo tanto, utilizado en fases posteriores de un estudio cuando un número limitado de factores están bajo investigación (Neter et al., 1996).

Este método se aplica típicamente a modelos grandes. Se usa a menudo como un paso previo a la aplicación de técnicas que requieren muchas evaluaciones del modelo, como la simulación del método Monte Carlo.

- MII (Mutual Information Index / Índice de Información Mutua): tiene como objetivo producir una medida de la información sobre la salida que se proporciona por una determinada entrada. La medida de sensibilidad se calcula sobre la base de un análisis probabilístico condicional.

Este método se usa para modelos con resultados dicotómicos, aunque también puede ser utilizado para salidas que son continuas (Critchfield & Willard, 1986). Es informativo y debido a la simplificación de las aproximaciones que pueden utilizarse en MII, la solidez de la clasificación basada en la medida de la sensibilidad puede ser difícil de evaluar.

- Regresión Logística**:** es un modelo de regresión para variables dependientes o de respuesta binomialmente distribuidas. Es útil para modelar la probabilidad de un evento ocurriendo como función de otros factores. Es un modelo lineal generalizado que se usa como función de enlace. (7)

Si hay separación entre los grupos, no hay solución única de la función de verosimilitud. Envuelve una gran cantidad de cómputo. Cuando hay distribución normal no se clasifica tan eficientemente como la función discriminante lineal.

- Sobol: es un método global basado en la descomposición de la varianza (Sobol, 1993). Este consiste en identificar un subconjunto de factores k que puedan explicar la mayor parte de la salida. El método presenta ventajas respecto de otras técnicas, en particular el hecho de que el cálculo de los índices es independiente del tipo de modelo, y por lo tanto, pueden estudiarse relaciones no-lineales. Para estimar los índices se utiliza un procedimiento numérico simplificado basado en simulaciones de Monte-Carlo, como alternativa para el cálculo de las integrales multidimensionales involucradas.

La mayor desventaja del método es el requerimiento de un gran número de simulaciones, lo que se traduce en un alto costo computacional cuando se desean analizar varios parámetros en forma simultánea. El método de Sobol tiene un costo computacional del orden de n (2k+1).

- Mínimo cuadrado: Permite encontrar la ecuación de una recta a partir de los datos experimentales, es decir, utilizando solamente las mediciones experimentales se obtendrá la pendiente y la ordenada al origen de la recta que mejor se ajuste a tales mediciones. Es un método objetivo, solo depende de los resultados experimentales. Proporciona una estimación probabilística de la ecuación que representa unos datos experimentales y proporciona intervalos pequeños de error. (8)

Actualmente se han desarrollado innumerables aplicaciones basadas en la minimización de una norma cuadrática en diversos campos que tienen relación con procesamiento de datos estadísticos o experimentales.
Las principales aplicaciones se agrupan en:

- Aproximación de funciones
- Estimación de parámetros (9)

1.4 Modelos matemáticos

Un modelo matemático se define desde el punto de vista de las matemáticas, como una descripción de un hecho o fenómeno del mundo real, para expresar relaciones que permiten estudiar comportamientos de sistemas complejos ante situaciones difíciles de observar en la realidad. (10)

Su construcción incluye:

1. Encontrar un problema del mundo real.
2. Identificar las posibles variables (dependientes e independientes) que intervienen en el problema y sus relaciones, para establecer hipótesis que puedan tratarse de manera matemática.
3. Aplicar los conocimientos que se posee para llegar a conclusiones matemáticas.

Es importante mencionar que un modelo matemático no es completamente exacto con problemas de la vida real, de hecho, se trata de una idealización. (11)

Dentro de los modelos matemáticos que más se utilizan se encuentran:

- Las Redes Petri: son herramientas para el modelado de sistemas de información que son considerados no determinísticos, concurrentes, paralelos, asíncronos, distribuidos y/o estocásticos, además son utilizadas para modelar el comportamiento dinámico de sistemas discretos. (12)

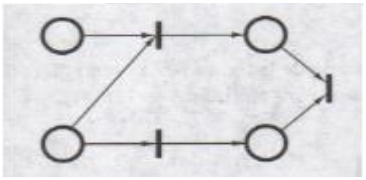

Figura 1: Redes Petri

Estas redes generales no pueden modelar ciertas situaciones de prioridad. En general, los problemas modelados con redes Petri, tienen altos costos de CPU y recursos del sistema.

- Las Redes Bayesianas: constituyen un método para la representación de conocimiento incierto, que permite establecer razonamientos basados en la teoría de la probabilidad. Formalmente, una red bayesiana es un grafo dirigido acíclico, con una distribución de probabilidad sobre sus variables, que satisface la condición de Markov, la cual plantea que la probabilidad de cualquier variable V, una vez que han sido determinados los valores de todos sus padres, es independiente de las variables que no son descendientes de V en la red. (13)

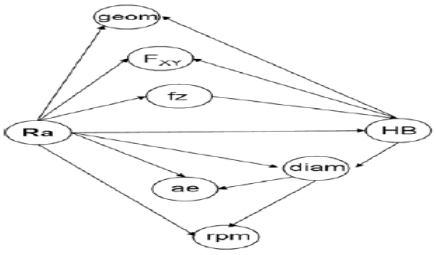

Figura 2: Redes Bayesianas

Uno de los principales problemas encontrados en este tipo de métodos son las posibles cantidades de estructurasen la que es dividida la red; es por ello que se han desarrollado diferentes heurísticas para no explorar todo el espacio de búsqueda, entre ellos figuran algoritmos genéticos y algoritmo “greedy” como el K2. (14)

- Los sistemas de ecuaciones diferenciales: son un conjunto de ecuaciones con varias funciones incógnitas y un conjunto de condiciones de contorno. Una solución de estos sistemas es un conjunto de funciones diferenciales que satisfacen todas y cada una de las ecuaciones del sistema. (15)

$$\frac{dx}{dt} = ax - bxy$$
$$\frac{dy}{dt} = -cy + oxy$$

Figura 3: Ejemplo de un sistema de ecuaciones diferenciales

La enorme importancia de estas ecuaciones en las matemáticas, y especialmente en sus aplicaciones, se debe principalmente al hecho de que la investigación de muchos problemas de ciencia y tecnología puede reducirse a la solución de tales ecuaciones. Los cálculos que requiere la construcción de maquinaria eléctrica o de dispositivos radiotécnicos, el cálculo de trayectorias de proyectiles, la investigación de la estabilidad de aeronaves en vuelo o del curso de una reacción química, todo ello depende de la solución de ecuaciones diferenciales. (16)

Por la utilidad de este tipo de modelos, es que se ha utilizado en la plataforma BioSyS para la representación de los problemas biológicos.

1.5 Aplicaciones existentes

El Análisis de Sensibilidad se ha aplicado en varios campos incluyendo sistemas de ingeniería compleja, Física, Química, Economía, en las aplicaciones financieras, análisis de riesgos, procesos de señales, redes neuronales, Biomecánica, Paleobiología, Geología, ciencias sociales, la toma de decisiones médicas, en la agricultura y cualquier área donde se desarrollen modelos. (17)

Sin embargo, en la revisión de aplicaciones publicadas para la Biología de Sistemas que realizarán esta técnica, los resultados arrojados fueron muy pocos, obteniéndose cuatro programas con desventajas en documentación, licencias, limitaciones y forma de almacenamiento de la información, que frenan en cada caso los estudios a realizarse:

- Berkeley Madonna v8.0.1**:** programa para analizar y modelar la dinámica de sistemas. Es uno

de los programas más utilizados para resolver ecuaciones diferenciales de propósitos generales disponibles en la actualidad. Desarrollado en Berkeley bajo el patrocinio de NSF[1] y NIH[2], se usa actualmente con fines académicos y comerciales en la construcción de modelos matemáticos. Entre las funciones que lo destacan se encuentran la construcción del modelo matemático (a partir del conocimiento de las ecuaciones, o a partir del conocimiento de las reacciones químicas que intervienen en el sistema en estudio) y el trabajo con parámetros (ajuste de curvas, optimización, Análisis de Sensibilidad y graficación).

La versión académica es totalmente funcional con las excepciones siguientes: los modelos no se guardan, las gráficas y tablas no se guardan ni se pueden copiar, el diálogo del Registro aparece cada vez que se inicie el programa y solo puede ser usada por una máquina a la vez. Si se desea para una comunidad de investigadores, se debe pagar la versión comercial, que si permite exportar las gráficas y tablas en formato de mapa de bit (.bmp) y las ecuaciones del modelo matemático, en formato de texto. Sin embargo, solamente corre en los sistemas operativos de Windows y Mac, aunque en la página oficial del software, se habla que están en desarrollo de una nueva versión en Java para el sistema operativo Linux. (18)

- EASY-FIT Express 1.0**:** software para la estimación de parámetros y el diseño experimental. Brinda grandes facilidades para la creación de reportes y gráficas, planos en tres dimensiones (3D) para variables de dos modelos (ajuste de superficies), entrada y salida de datos desde y para archivos de texto y Microsoft Excel.

  El software es gratuito, pero presenta algunas restricciones de uso: 30 criterios de ajuste y de conjuntos de datos, 100 variables de modelado, 2,000 valores de tiempo, 3,000 datos experimentales, 200 restricciones lineales o no lineales. Implementado para el sistema operativo Microsoft Windows XP o Vista. (19)

- XPPAUT**:** es un programa para resolver ecuaciones diferenciales. Aprovecha las ventajas del programa XPP (resolver ecuaciones diferenciales) y el programa AUTO (bifurcación de poblaciones) para validar la salida de uno en el otro. Es gratuito y corre en los sistemas operativos de Windows, MacOS X, Linux/Unix y otros. Permite exportar las ecuaciones,

[1] NSF: National Science Foundation. Agencia del gobierno de Estados Unidos que impulsa investigación y educación fundamental, en todos los campos no médicos de la Ciencia y la Ingeniería

[2] NIH: National Institutes of Health. Agencia del gobierno norteamericano de salud y servicios humanos

parámetros, datos iniciales, y las simulaciones para un fichero de texto siendo bastante engorrosa su lectura. Además de que lamentablemente existe muy poca documentación al respecto. (20)

- COPASI**:** es el software más completo para realizar cualquiera de las técnicas estudiadas en este trabajo. Permite diseñar experimentos para estimar parámetros, el Análisis de Sensibilidad de los mismos, y estimar los parámetros desconocidos. Almacena los datos en un formato específico de COPASI aunque permite exportar tanto para los programas Madonna y XPPAUT descritos anteriormente. Pero tiene la deficiencia de poseer licencias comerciales y restrictivas (no comercial). La licencia no comercial justifica el uso del software solo en caso de estudios internos de empresas sin fines comerciales (ejercicios académicos o programas gratuitos). Esto claramente frena los estudios a realizarse por parte de los especialistas, ya que tienen como objetivo principal la obtención de productos tanto como para el insumo nacional como la exportación extranjera. (21)

De manera general puede concluirse que los clientes del CIM necesitan de una versión propia que personalice y cubra, sus necesidades y requerimientos, ya que estos presentan desventajas en documentación, licencias, limitaciones y forma de almacenamiento de la información, que frenan en cada caso los estudios a realizarse por el investigador.

1.6 Metodología de desarrollo de software

Es un conjunto de pasos y procedimientos que deben seguirse para desarrollar software. Utilizada para dividir un proyecto en etapas, desarrollando las tareas que se llevan a cabo en cada etapa y las restricciones que deben aplicarse en cada una de ellas. En estas metodologías se especifican las técnicas y herramientas que se deben emplear para controlar y gestionar un proyecto. (22)

Actualmente, existen diferentes metodologías y técnicas para el desarrollo de productos las que fueron analizadas y definidas en la arquitectura del proyecto, quedando como propuesta final la metodología OpenUP. (23)

1.6.1 OpenUP

OpenUP/Basic es un Framework de procesos de desarrollo de programas de código abierto, que con el tiempo espera cubrir un amplio conjunto de necesidades en el campo del desarrollo de programas. Permite un abordaje ágil al programa en análisis con sólo proveer un conjunto simplificado de contenidos, fundamentalmente relacionados con orientación, productos de trabajo, roles, y tareas. Es un proceso interactivo de desarrollo de software simplificado, completo y extensible; para pequeños equipos de desarrollo, que valoran los beneficios de la colaboración y de los involucrados, con el resultado del proyecto, por encima de formalidades innecesarias. (24)

Esta metodología es la utilizada en el proyecto ya que permite detectar errores tempranos a través de un ciclo iterativo, y también por ser una metodología ágil, que tiene un enfoque centrado en el cliente.

- **Ciclo de vida OpenUP**

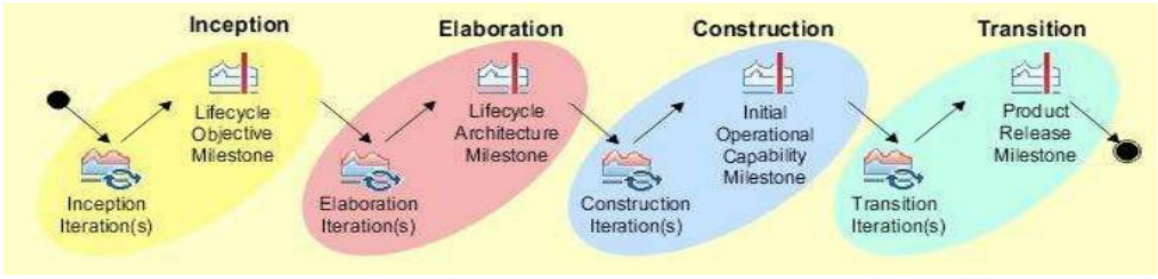

Figura 4: Ciclo de vida OpenUP

En la figura cuatro se puede observar las cuatro fases de desarrollo que propone la metodología: la primera es la fase de Concepción, en donde los objetivos principales son:

- Realizar el planteamiento del problema.
- Elaborar la justificación.
- Definir objetivo general y objetivos específicos.
- Obtener toda la información relacionada con el proyecto.
- Capturar las necesidades de los clientes en los objetivos del ciclo de vida para el proyecto.

La segunda fase es la de Elaboración. Esta fase tiene como objetivo:

- Definir los riesgos significativos para la arquitectura.
- Establecer la base para la elaboración de la arquitectura del sistema.

La tercera fase es Construcción. Esta fase tiene como objetivo:

- Diseñar, implementar y realizar las pruebas de las funcionalidades para realizar un sistema completo.
- Completar el desarrollo del sistema basado en la arquitectura definida.

La última fase es la de Transición. Esta fase tiene como objetivo:

- Asegurar que el sistema sea entregado a los usuarios.
- Evaluar la funcionalidad y escena del último entregable de la fase de construcción.

1.7 Herramientas CASE

Las herramientas CASE (Computer Aided Software Engineering / Ingeniería de Software Asistida por Computadora) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software. (25) Reducen el costo de las mismas en términos de tiempo y de dinero, ya que permiten la generación automática de código, documentación o detección de errores.

1.7.1 Visual Paradigm

Es una herramienta UML[3] profesional que soporta el ciclo de vida completo de desarrollo del software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El lenguaje de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, y generación automática de código. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación.

Soporta los principales estándares de la industria tales como SysML, BPMN, XMI, entre otros. Ofrece un completo conjunto de herramientas de los equipos de desarrollo de software necesario

[3] UML: Lenguaje Unificado de Modelado (por sus siglas en inglés, Unified Modeling Language)

para los requisitos de la captura, software de planificación, la planificación de controles, el modelado de clases, modelado de datos, entre otros. (26)

1.8 Lenguajes de programación

Son herramientas que permiten crear programas y software. Entre ellos se tiene Delphi, Visual Basic, Java, etc. Los lenguajes de programación facilitan la tarea de programación, ya que disponen de formas simbólicas y en manera de un texto los códigos que podrán ser leídos y escritos por personas, a su vez resultan independientes del modelo de computador a utilizar. (27)

1.8.1 Lenguaje Java

Java es un lenguaje de programación desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria. Una de las principales características que favoreció el crecimiento y difusión del lenguaje Java es su capacidad de que el código funcione sobre cualquier plataforma de software y hardware.

Este lenguaje tiene otras características importantes entre las que se encuentran:

- 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 estos.
- 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 las posibilidades de error. Maneja la memoria para eliminar las preocupaciones por parte del programador de la liberación o corrupción de la misma.

- Seguro: el código de Java puede ser ejecutado en un entorno que prohíbe la introducción de virus, borrar y modificar ficheros o la ejecución de operaciones que provoquen la caída del ordenador y la pérdida de datos.

1.9 Herramientas de desarrollo

Las herramientas de desarrollo son aquellos programas o aplicaciones que tengan cierta importancia en el desarrollo de un programa (programación). Pueden ser de importancia vital (como un ensamblador, un compilador o un editor) o de importancia secundaria, como una IDE (Integrated Development Environment - Entorno de Desarrollo Integrado).

1.9.1 IDE de desarrollo NetBeans

Es una aplicación de código abierto (open source) diseñada para el desarrollo de aplicaciones, fácilmente portables entre las distintas plataformas (uso de la tecnología Java y un entorno de desarrollo integrado). Dispone de soporte para crear interfaces gráficas de forma visual, desarrollo de aplicaciones web, control de versiones, entre otras.

Es un programa con licencia CDDL[4] que permite su uso libremente, brinda el código del mismo y es gratuita. Permite escribir, compilar, hacer un debug, ensamblar y desplegar aplicaciones, y aunque está escrito en Java, brinda soporte para toda clase de lenguajes de programación. Además, funciona en sistemas operativos compatibles con la máquina virtual Java (Windows XP, Vista, Windows 7, Ubuntu 9.10, Solaris, Mac OS X 10.5 o superior, entre otros). (28)

1.10 Arquitectura de software

En la actualidad existen varias definiciones de Arquitectura de software (AS), pero una de las más aceptadas es la que propone Clements: La AS es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se la percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema. La vista arquitectónica es una vista abstracta, aportando el más alto nivel de comprensión y la supresión o diferimiento del detalle inherente a la

[4] CDDL: Common Development and Distribution License o Licencia Común de Desarrollo y Distribución, en español

mayor parte de las abstracciones (29)

1.10.1 Modelo Vista Controlador

Es un estilo de programación en el que el objetivo primordial es la separación de la lógica de negocio de la lógica de diseño. La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en tres capas: la de presentación, la de negocio y la de datos, en caso de que sobrevenga algún cambio, sólo se afecta la capa requerida.

- Capa de presentación: es la que le permite al usuario interactuar con el sistema. Esta capa se encarga de capturar y mostrar la información del usuario; y se comunica únicamente con la capa de negocio.
- Capa de negocio: es donde residen los programas que se ejecutan. En esta capa es donde se reciben y envían las peticiones de los usuarios, además, es donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él. También se consideran aquí los programas de aplicación.
- Capa de datos: es donde residen los datos. Esta capa está formada por uno o más gestores de bases de datos que reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio. (30)

1.11 Conclusiones del capítulo

A lo largo de este capítulo se ha explicado y detallado la base sobre la que se trabajará en aras de desarrollar la aplicación. Del estudio se definió lo siguiente: trabajar con modelos matemáticos formados por sistemas de ecuaciones diferenciales, utilizar la función de Mínimo Cuadrado para la realización del Análisis de Sensibilidad, usar Java como lenguaje de programación, como IDE de desarrollo NetBeans, como herramienta de modelado Visual Paradigm y OpenUP como metodología de desarrollo.

Capítulo 2: Diseño del Sistema

2. Introducción

En este capítulo se realiza el diseño de la aplicación basada en la metodología OpenUp, la cual establece primeramente la definición de los actores del sistema, posteriormente los requisitos funcionales para la modelación del Diagrama de Casos de Usos del Sistema y, los requisitos no funcionales. Finalmente propone la modelación del Diagrama de Clases del Diseño y el Diagrama de Interacción por cada caso de uso.

2.1 Actores del Sistema

Los actores de un sistema pueden ser sistemas, hardware externo o personas que interactúan con el sistema, ya sea para inicializar una funcionalidad o brindarle información a este.

Nombre del Actor	Descripción
Investigador	El Investigador o científico del polo cubano es aquella persona que va a interactuar con el sistema para solicitar el análisis de sensibilidad.

Tabla 1: Actores del Sistema

2.2 Requisitos funcionales

Son capacidades o condiciones que el sistema debe cumplir. Se mantienen invariables sin importar con qué propiedades o cualidades se relacionen.

CU1. Gestionar Parámetros Conocidos.

RF1. Insertar Parámetros Conocidos.

RF2. Cargar Parámetros Conocidos.

RF3. Modificar Parámetros Conocidos.

RF4. Eliminar Parámetros Conocidos.

CU2. Gestionar Parámetros Desconocidos

RF5. Insertar Parámetros Desconocidos.

RF6. Cargar Parámetros Desconocidos.

RF7. Modificar Parámetros Desconocidos.

RF8. Eliminar Parámetros Desconocidos.

CU3. Determinar Sensibilidad de los Parámetros.

RF9. Determinar Sensibilidad de los Parámetros.

2.3 Requisitos no Funcionales

Restricciones que afectan a los servicios o funciones del sistema, tales como restricciones de tiempo, definición de estándares y otras condiciones necesarias para que el sistema se encuentre funcional. Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Estas propiedades son las que hacen al producto atractivo, usable, rápido o confiable.

1. Apariencia o interfaz externa

   RNF 1. La aplicación deberá tener una interfaz externa amigable, que sea sencilla y fácil de entender por el usuario.

2. Software

   RNF 2. La máquina virtual de Java 1.5 tiene que estar instalada en la PC.

   RNF 3. Puede instalarse sobre cualquier sistema operativo (Multiplataforma)

3. Hardware

RNF 4. Se debe contar con 256 MB de memoria RAM como mínimo

4. Requisitos Legales, de Derecho de Autor y otros

RNF 5. El sistema debe ser llevado al CIM mediante un proceso de transferencia, una vez que esté en explotación, incluyendo el código fuente y la documentación correspondiente.

RNF 6. No se hace ninguna solicitud de derecho de autor, patentes, marca comercial o complacencia con logotipo para el software, debido a que se usan soluciones con Licencia Pública General (GNU GPL por sus siglas en inglés), bajo el principio de software libre.

5. Usabilidad

RNF 7. La aplicación dará la posibilidad de almacenar los resultados obtenidos mediante interfaces que permitan el intercambio de datos de manera fácil para el usuario.

6. Soporte

RNF 8. Terminado el software se prestarán los servicios de instalación y configuración de la aplicación.

RNF 9. Se prestarán servicios de mantenimiento del software.

7. Confidencialidad

RNF 10. La información manejada por el sistema está protegida de acceso no autorizado y divulgación.

2.4 Diagrama de Casos de Usos del Sistema

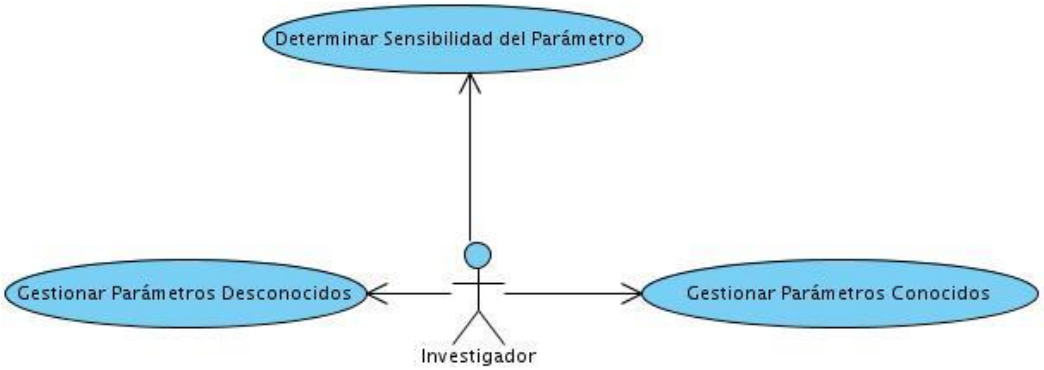

Figura 5: Diagrama de Casos de Usos del Sistema

2.5 Descripción de los Casos de Usos del Sistema

Los casos de usos del sistema y los requisitos funcionales de entrada y salida de la solución a implementar, constituyen la entrada fundamental para el diseño del análisis de sensibilidad en la plataforma BioSyS.

2.5.1 Caso de Uso Gestionar Parámetros Conocidos

Caso de Uso:	Gestionar Parámetros Conocidos
Tipo:	Funcional
Actores:	Investigador
Resumen:	El CU inicia cuando el Investigador necesita "Gestionar Parámetros Conocidos", para lo cual el sistema va a permitir insertar, cargar, modificar o eliminar los parámetros que se conocen dentro de un modelo matemático. El CU termina una vez que el Investigador concluya de gestionar algún parámetro conocido.
Precondiciones:	El Investigador tiene que estar autenticado. El modelo matemático tiene que estar cargado.
Referencias	RF1, RF2, RF3 y RF4
Prioridad	Crítico
Complejidad	

Flujo Normal de Eventos	
Acción del Actor	**Respuesta Sistema**
1. El Investigador realiza una de las siguientes opciones: • Insertar Parámetros Conocidos. • Modificar Parámetros Conocidos. • Cargar Parámetros Conocidos. • Eliminar Parámetros Conocidos.	2. El sistema muestra la interfaz correspondiente a la acción que va a realizar el usuario.
Escenario: "Insertar Parámetros Conocidos"	
Acción del Actor	**Respuesta Sistema**
	3. El sistema muestra la interfaz de análisis de sensibilidad.
4. El Investigador introduce los valores de necesarios del análisis.	5. El sistema valida que los datos entrados sean correctos.
6. El investigador escoge la opción de insertar en la base de datos.	7. El sistema guarda los datos en la base de datos.
Prototipo de Interfaz	
Flujos Alternos	
Acción	**Respuesta del Sistema**
	5.1 El sistema muestra algunos de los siguientes mensaje de error : • Debe variar un parámetro • Entre correctamente los valores de los

	parámetros. • Entre el tiempo inicial correctamente. • Entre el tiempo final correctamente. • Entre el error permitido correctamente.
4.1 Si el investigador no introduce los valores, finaliza el caso de uso. 4.2 El investigador introduce los valores que faltan correctamente.	
Prototipo de Interfaz	
Escenario: "Cargar Parámetros Conocidos"	
Acción	**Respuesta del Sistema**
	1. El sistema muestra la interfaz para cargar los parámetros conocidos.
2. El Investigador selecciona el parámetro que desea cargar.	3. El sistema muestra una interfaz con el parámetro seleccionado cargado.
Prototipo de Interfaz	
Escenario: "Modificar Parámetros Conocidos"	
Acción	**Respuesta del Sistema**
	1. El sistema muestra la interfaz de análisis de sensibilidad con los parámetros conocidos.
2. El investigador selecciona los parámetros conocidos que desea modificar.	3. El sistema guarda los cambios realizado en la base

	de datos.
4. El investigador cierra la interfaz de análisis de sensibilidad.	
Prototipo de Interfaz	
Escenario: "Eliminar Parámetros Conocidos"	
Acción	**Respuesta del Sistema**
1. El investigador selecciona el parámetro conocido que desea eliminar.	2. El sistema muestra un mensaje de alerta "Está seguro que desea eliminar el parámetro".
3. El investigador confirma aceptando.	4. El sistema elimina el parámetro de la base de datos.
Prototipo de Interfaz	
Poscondiciones	El parámetro conocido queda insertado, modificado, cargado o eliminado.

Tabla 2: Gestionar Parámetros Conocidos

2.5.2 Caso de Uso Gestionar Parámetros Desconocidos

Caso de Uso:	Gestionar Parámetros Desconocidos
Tipo:	Funcional
Actores:	Investigador
Resumen:	El CU inicia cuando el Investigador necesita "Gestionar Parámetros Desconocidos", para lo cual el sistema va a permitir insertar, cargar, modificar o eliminar los parámetros que se desconocen dentro de un modelo matemático. El CU

	termina una vez que el Investigador concluya de gestionar algún parámetro desconocido.
Precondiciones:	El Investigador tiene que estar autenticado. El modelo matemático tiene que estar cargado.
Referencias	RF5, RF6, RF7 y RF8
Prioridad	Crítico
Complejidad	

Flujo Normal de Eventos	
Acción del Actor	**Respuesta Sistema**
1. El Investigador realiza una de las siguientes opciones: 1 Insertar Parámetros Desconocido. 2 Modificar Parámetros Desconocido. 3 Cargar Parámetros Desconocido. 4 Eliminar Parámetros Desconocido.	2. El sistema muestra la interfaz correspondiente a la acción que va a realizar el usuario.
Escenario: "Insertar Parámetros Desconocido"	
Acción del Actor	**Respuesta Sistema**
	3. El sistema muestra la interfaz de análisis de sensibilidad.
4. El Investigador introduce los valores de necesarios del análisis.	5. El sistema valida que los datos entrados sean correctos.
6. El investigador escoge la opción de insertar en la base de datos.	7. El sistema guarda los datos en la base de datos.
Prototipo de Interfaz	
Flujos Alternos	
Acción	**Respuesta del Sistema**

	5.1 El sistema muestra algunos de los siguientes mensaje de error : • Debe variar un parámetro • Entre correctamente los valores de los parámetros. • Entre el tiempo inicial correctamente. • Entre el tiempo final correctamente. • Entre el error permitido correctamente.
4.1 Si el investigador no introduce los valores, finaliza el caso de uso. 4.2 El investigador introduce los valores que faltan correctamente.	
Prototipo de Interfaz	
Escenario: "Cargar Parámetros Desconocidos"	
Acción	**Respuesta del Sistema**
	4. El sistema muestra la interfaz para cargar los parámetros desconocidos.
5. El Investigador selecciona el parámetro que desea cargar.	6. El sistema muestra una interfaz con el parámetro seleccionado cargado.
Prototipo de Interfaz	
Escenario: "Modificar Parámetros Desconocidos"	
Acción	**Respuesta del Sistema**

	5. El sistema muestra la interfaz de análisis de sensibilidad con los parámetros desconocidos.
6. El investigador selecciona los parámetros desconocidos que desea modificar.	7. El sistema guarda los cambios realizado en la base de datos.
8. El investigador cierra la interfaz de análisis de sensibilidad.	
Prototipo de Interfaz	
Escenario: "Eliminar Parámetros Desconocido"	
Acción	**Respuesta del Sistema**
5. El investigador selecciona el parámetro desconocido que desea eliminar.	6. El sistema muestra un mensaje de alerta "Está seguro que desea eliminar el parámetro".
7. El investigador confirma aceptando.	8. El sistema elimina el parámetro de la base de datos.
Prototipo de Interfaz	
Poscondiciones	El parámetro desconocido queda insertado, modificado, cargado o eliminado.

Tabla 3: Gestionar Parámetros Desconocidos

2.5.3 Caso de Uso Determinar Sensibilidad del Parámetro

Caso de Uso:	Determinar Sensibilidad de los Parámetros
Tipo:	Funcional

Actores:	Investigador
Resumen:	El caso de uso comienza cuando el Investigador desea "Determinar Sensibilidad del Parámetro", el sistema evalúa la sensibilidad del parámetro comparando dos distribuciones de valores del parámetro asociadas a los resultados de sensibles o no sensibles. El caso de uso termina una vez mostrado el resultado.
Precondiciones:	El Investigador debe estar autenticado.
Referencias	RF9
Prioridad	Crítico
Complejidad	

Flujo Normal de Eventos	
Acción del Actor	**Respuesta Sistema**
1. El Investigador selecciona en la aplicación la opción de realizar análisis de sensibilidad.	2. El sistema muestra la interfaz correspondiente a análisis de sensibilidad.
3. El Investigador introduce los parámetros a los cuales se les va a realizar el análisis.	4. El sistema valida los datos.
5. Selecciona el o los parámetros que va a variar para fijar el rango donde se encuentran los valores.	6. El sistema muestra una tabla con el valor obtenido en las simulaciones para cada parámetro variado y si es sensible o no sensible.

Escenario: Graficar Resultados	
Acción del Actor	**Respuesta Sistema**
1. A partir de la tabla generada se pueden seleccionar las opciones: a) View Tree b) View clusters	2. El sistema muestra la gráfica correspondiente a la acción del actor.

Prototipo de Interfaz	
Flujos Alternos	
Acción	**Respuesta del Sistema**
	4.1 El sistema muestra algunos de los siguientes mensaje de error : • Debe variar un parámetro • Entre correctamente los valores de los parámetros. • Entre el tiempo inicial correctamente. • Entre el tiempo final correctamente. • Entre el error permitido correctamente.
3.1 Si el investigador no introduce los valores, finaliza el caso de uso. 3.2 El investigador introduce los valores que faltan correctamente.	
Poscondiciones	Se muestra el resultado de la sensibilidad del parámetro.

Tabla 4: Determinar Sensibilidad del Parámetro

2.6 Diagrama de Clases del Diseño

Representa las clases que serán utilizadas dentro del sistema y las relaciones que existen entre ellas. Sirve para visualizar las relaciones entre las clases que involucran el sistema, las cuales pueden ser asociativas, de herencia, de uso y de convencimiento. Un diagrama de clases está compuesto por los siguientes elementos: Clase: atributos, métodos y visibilidad; y Relaciones:

Herencia, Composición, Agregación, Asociación y Uso. (31)

En la figura seis, se muestran las siete clases definidas en la propuesta de solución: La clase BSSensivityAnalysisView, es la clase encargada de recoger todos los parámetros de entrada al sistema. Esta clase va a relacionarse con la clase BSControlSensitivityAnalysis para realizar el análisis del modelo. Por su parte la clase BSControlSensitivityAnalysis va a implementar los métodos definidos en BSSensitivityAnalysis, la que a su vez va a estar relacionada con el subsistema Weka para graficar la solución.

Los paneles ParametersConditionsTable y BSOdeSetPanel están incluidos en la vista para recoger la información que se introduce en los componentes.

Las clases BSParametersConditionManager y BSOdeSetPanelManager son las encargadas de procesar la información entrada en los paneles incluidos en la vista.

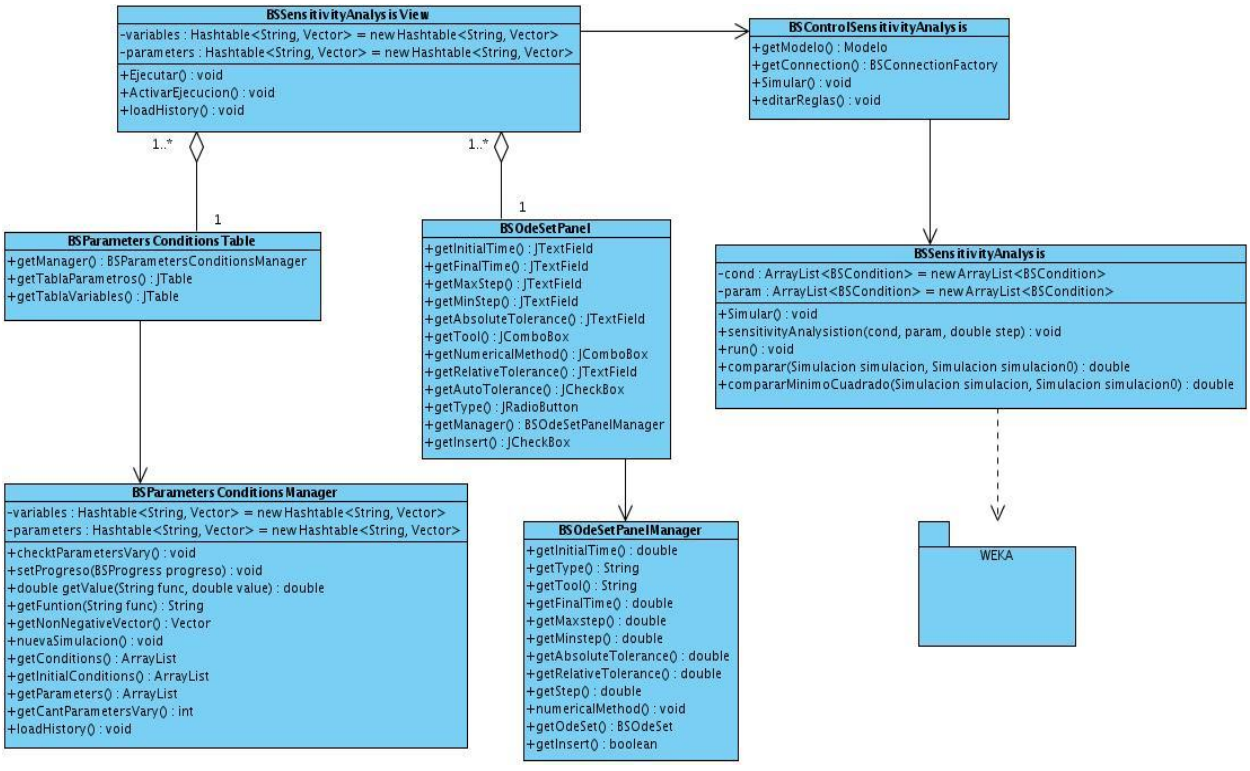

Figura 6: Diagrama de Clases del Diseño

2.7 Diagrama de Vista Lógica

Representa un subconjunto del artefacto modelo de diseño. Éste describe las clases más importantes, su organización en paquetes y subsistemas y éstos a su vez en capas siempre que sea posible.

En la figura siete, se muestran las clases organizadas, haciendo uso del Modelo Vista Controlador, el cuál es un estilo de programación en el que el objetivo primordial es la separación de la lógica de negocio de la lógica de diseño. En el paquete Vistas se encuentran las clases con la que el usuario interactúa, en el paquete Modelo se encuentra las clases que van a tener implementadas toda la lógica del negocio y en el subsistema BSConnectionFactory se encuentran las clases que posibilitan el acceso a la base de datos mediante el framework Hibernate.

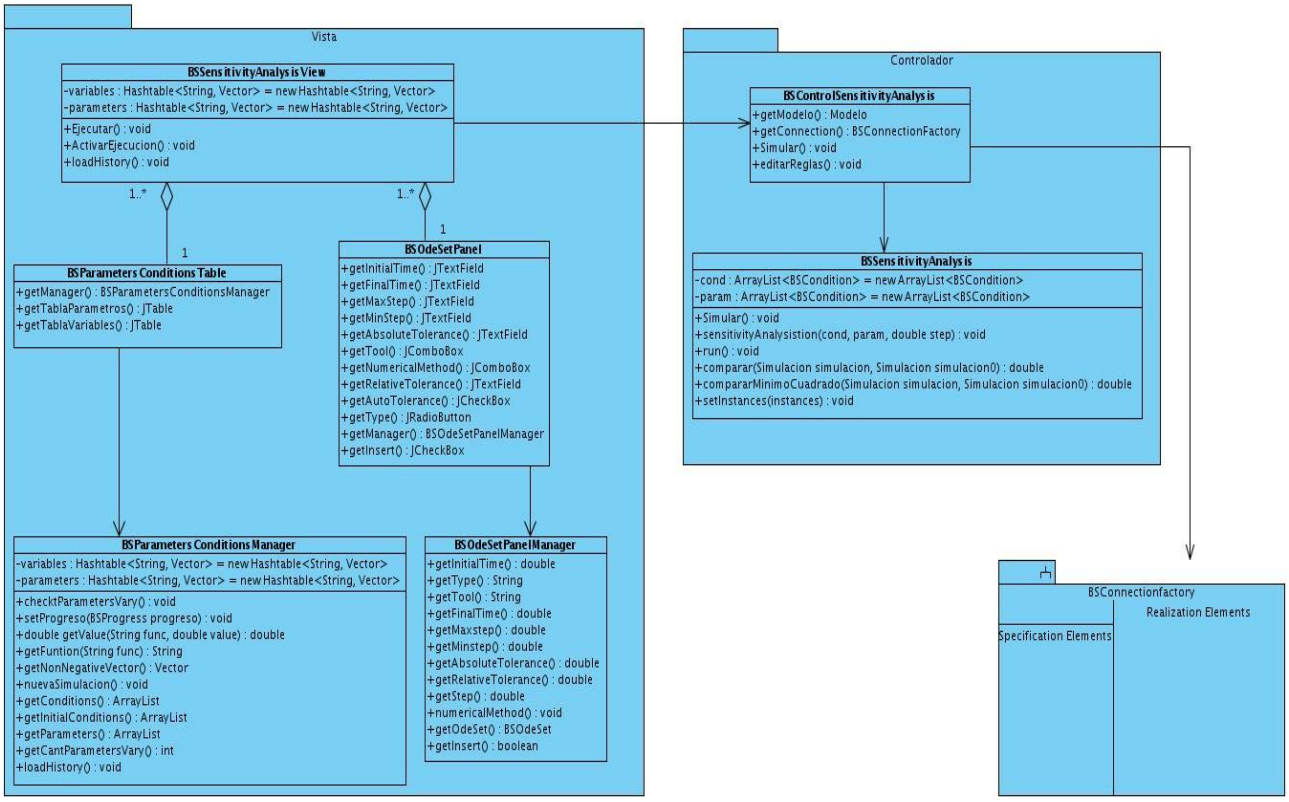

Figura 7: Diagrama de Vista Lógica

2.8 Diagramas de interacción

Un diagrama de interacción 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 de un sistema. Ambos diagramas (secuencia y colaboración) son semánticamente equivalentes. Se puede pasar de uno a otro sin pérdida de información y se utilizan para modelar los aspectos dinámicos de un sistema.

2.8.1 Diagrama de secuencia

Muestran gráficamente las interacciones del actor y de las operaciones a que dan origen. El diagrama de secuencia muestra un determinado escenario de un caso de uso, los eventos generados por actores externos, su orden y los eventos internos del sistema. Estos destacan la ordenación temporal de los mensajes.

En el trabajo se definió un diagrama de secuencia por cada uno de los requisitos funcionales que componen cada caso de uso (ver **Anexo 1**). La figura ocho muestra cómo quedaría el CU Determinar Sensibilidad de los Parámetros.

En la figura ocho se muestra el diagrama de secuencia, correspondiente al caso de uso Determinar Sensibilidad del Parámetro. Donde la clase BSControlSensitivityAnalysis visualiza la interfaz BSSensitivityAnalysisView, esta interfaz es la que se encarga de recoger los datos de los parámetros introducidos por el investigador, luego se resuelve el modelo matemático con estos datos mediante el método Simular(), una vez terminado el proceso de simulación los datos son introducidos en el método compararMinimoCuadrado(), este método va a determinar cuáles de los parámetros variados son sensibles y cuáles no, estos resultados son guardados en un archivo con extensión **.arff**, de donde WEKA obtiene los datos y los muestra gráficamente.

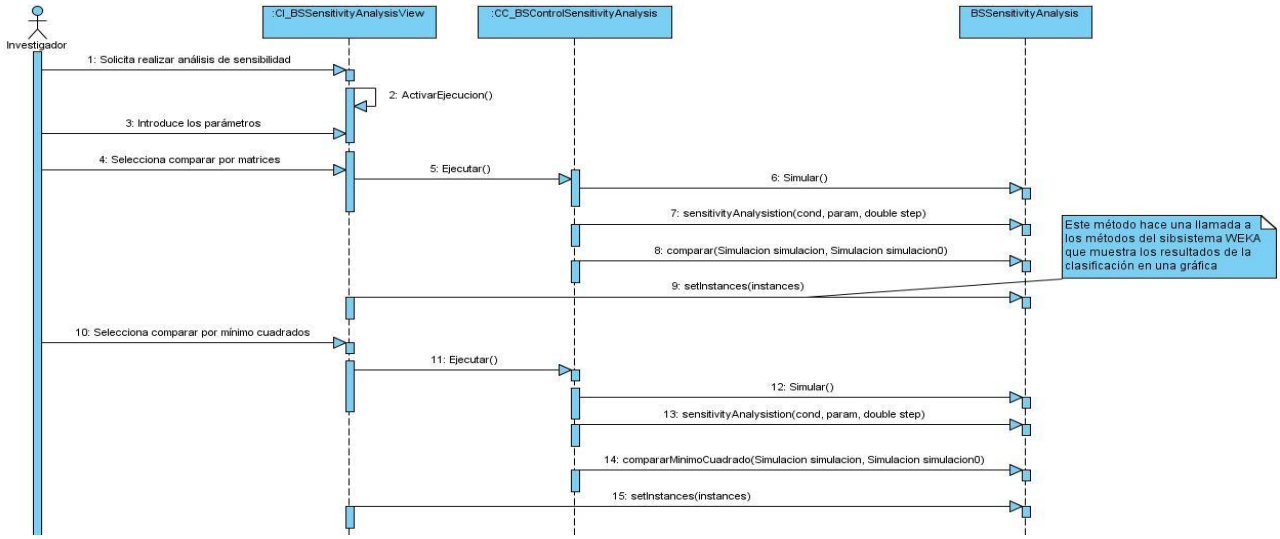

Figura 8: Determinar Sensibilidad de los Parámetros

2.9 Patrones de Diseño

Describe un problema que ocurre varias veces y también describe el núcleo de la solución al problema, de forma que puede utilizarse en ilimitadas ocasiones sin tener que hacer dos veces lo mismo.

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular. En otras palabras, un patrón de diseño no es más que una solución estándar para un problema común de programación. Una técnica para flexibilizar el código haciéndolo satisfacer ciertos criterios. (32)

2.9.1 Patrones GRASP

GRASP es un acrónimo de General Responsibility Assignment Software Patterns (patrones generales de software para asignar responsabilidades). El nombre se eligió para sugerir la importancia de aprender estos principios para diseñar con éxito el software orientado a objetos. Los patrones GRASP son utilizados para describir los principios fundamentales del diseño y la asignación de responsabilidades. (32)

- **Experto en Información**

Un Modelo de diseño podría definir cientos o miles de clases software, y una aplicación podría requerir que se realicen cientos o miles de responsabilidades. Durante el diseño de objetos, cuando se definen las interacciones entre los objetos, se toman decisiones sobre la asignación de responsabilidades a las clases. Haciendo un resumen de lo anteriormente planteado este patrón consiste en asignar la responsabilidad a la clase que tiene la información necesaria para realizarla.

En este caso la clase BSControlSensitivityAnalysis.java es la que maneja la información de los parámetros, por lo que la clase BSSensitivityAnalysisView.java se va a encargar de enviarle los datos para realizar el análisis de la sensibilidad de los parámetros. Evidenciándose de esta forma el patrón Experto en Información.

- **Creador**

La creación de instancias es una de las actividades más comunes en un sistema orientado a objetos. En consecuencia, es útil contar con un principio general para la asignación de las responsabilidades de creación. Si se asignan bien, el diseño puede soportar un bajo acoplamiento, mayor claridad y reutilización. El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos, una tarea muy común. La intención básica del patrón es encontrar un creador que necesite conectarse al objeto creado en alguna situación. Eligiéndolo como el creador se favorece el bajo acoplamiento.

En la aplicación se pone de manifiesto este patrón, cuando la clase BSControlSensitivityAnalysis.java crea una instancia de BSSensitivityAnalysis.java para invocar al método de realizar simulación.

```
BSSensitivityAnalysis control = new BSSensitivityAnalysis ();
control.Simular ();
```

- **Alta Cohesión**

La cohesión es la medida de la fuerza que une a las responsabilidades de una clase. Una clase con baja cohesión hace muchas cosas no relacionadas, o hace demasiado trabajo. Tales clases no son convenientes ya que son difíciles de mantener, de reutilizar y de entender. Una clase con

alta cohesión mejora la claridad y la facilidad de su uso, su mantenimiento se simplifica y es fácil de reutilizar.

Este patrón se evidencia en la aplicación ya que la clase BSControlSensitivityAnalysis, es la clase encargada de realizar la operación determinar sensibilidad del parámetro y de afectarse esta clase no afectaría las demás funcionalidades, por lo que se considera que tiene alta cohesión.

- **Bajo Acoplamiento**

El acoplamiento es una medida de la fuerza con que un elemento está conectado a otro, un elemento con bajo acoplamiento no depende demasiado de otros elementos. Una clase con alto acoplamiento confía en muchas otras clases, tales clases podrían no ser deseables ya que son muy complicadas de entender, son difíciles de reutilizar y los cambios realizados en las clases relacionadas fuerzan cambios locales.

Este patrón se evidencia dentro de la aplicación en la capa modelo ya que las clases de acceso a los datos tienen bastante independencia de las clases de abstracción de datos. Hay poca dependencia entre esas clases lo que permite una mayor reutilización.

2.9.2 Patrón DAO

Data Access Object / Objeto de Acceso a Datos, es un patrón de diseño que centraliza todo el acceso a datos en una capa independiente, aislándolo del resto de la aplicación. Su principal beneficio es que reduce la complejidad de los objetos de negocio al abstraerlos de la implementación real de la comunicación con la fuente de datos y de esta forma permitir una migración más fácil de fuente de datos. El uso de este patrón en el sistema se evidencia en la capa de Acceso a Datos (BSConnectionFactoy.class), ella se encarga de realizar la gestión de los datos entre las clases que contienen la lógica de negocio y las entidades.

2.10 Conclusiones del Capítulo

A lo largo de este capítulo se realizó el diseño de la aplicación. Se definieron los requisitos funcionales y no funcionales para el funcionamiento de la aplicación. Se modeló el Diagrama de Casos de Usos del Sistema, el Diagrama de Clases del Diseño y los Diagramas de interacción (secuencia) para una mejor comprensión de las funcionalidades del sistema. Por último, se aplicó el patrón GRASP para asignar responsabilidades a las clases y el patrón DAO para garantizar un acceso a datos orientado a objetos.

Capítulo 3: Implementación y Prueba del Sistema

3. Introducción

En este capítulo se diseñó el Diagramas Despliegue y el Diagrama de Componentes, en el cuál se representan los principales componentes y sus relaciones. También se realizó la implementación del sistema. Para ello se muestra el código fuente de los principales métodos y se realizan las pruebas necesarias para validar la solución.

3.1 Diagrama de Despliegue

El diagrama de despliegue es un tipo de diagrama del Lenguaje Unificado de Modelado que se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes. (33)

Los elementos usados por este tipo de diagrama son:

- Nodos: elementos de procesamiento con al menos un procesador, memoria, y posiblemente otros dispositivos.

- Dispositivos: nodos estereotipados sin capacidad de procesamiento en el nivel de abstracción que se modela.
- Conectores: expresan el tipo de conector o protocolo utilizado entre el resto de los elementos del modelo. (34)

En la figura nueve se muestra el diagrama de despliegue del sistema. En el cuál se aprecia:

- El Nodo PC Cliente: representa la máquina cliente asignada al investigador, que en este caso es el cliente del sistema y permite gestionar las peticiones del cliente. Este nodo estará conectado a un servidor de base de datos por medio del protocolo JDBC.
- El Nodo Servidor de BD: representa el servidor de base de datos utilizado para el almacenamiento de los datos de la aplicación.

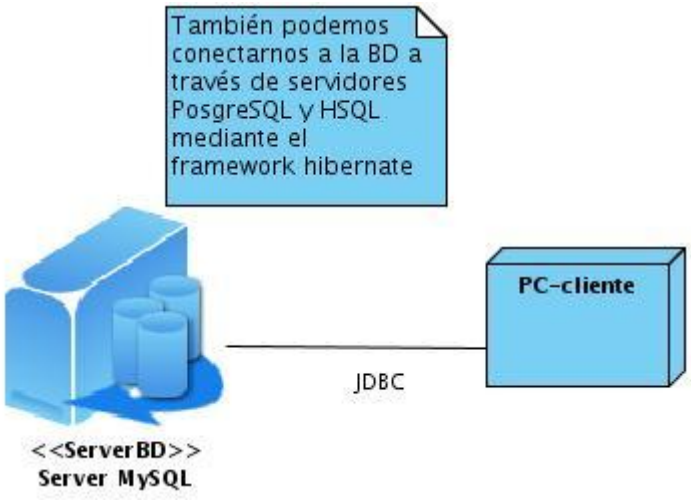

Figura 9: Diagrama de Despliegue

3.2 Diagrama de Componentes

El diagrama de componentes es un tipo de diagrama que muestra las organizaciones y dependencias lógicas entre los componentes del software (ya sean código fuente, binarios o ejecutables). (35)

Los elementos de modelado dentro de un diagrama de componentes serán:

- Componentes: incluyen librerías, tablas, archivos, ejecutables y documentos que formen parte del sistema.
- Paquetes: representan subsistemas.

En la figura diez se muestra el diagrama de componentes del sistema propuesto. Donde:

- El componente Visualizador Sensitivity.jar: permite mantener el control del sistema y la gestión de las funcionalidades del sistema por medio de interfaces, representa la cara principal del sistema, ya que las clases que contiene son las que permiten que se logre un buen funcionamiento del sistema, por sí solo no contiene funcionalidad, depende totalmente de los componentes.jar para lograr el resultado esperado.
- El componentes.jar: representan cada una de las funcionalidades del sistema, es decir los .jar correspondientes a cada uno de los módulos del sistema, la comunicación entre ellos se establece por medio de interfaces, así como con la interfaz principal.

- El componente DataSourse DAO: representa aquellos objetos de acceso a datos (Patrón DAO).
- El componente Base Datos: representa la base de datos física, a ella aceden todos los subsistemas por medio del API de Java JDBC, protocolo de comunicación que permite el acceso a los datos asociado al ORM Hibernate.
- Las interfaces: representan clases que contienen operaciones que son brindadas por los componentes simulación.jar y editor de ecuaciones.jar, utilizadas por el componente de análisis.jar.
- El componente Mysql-connector-java-3.1.11-bin y postgresql-8.3-603.jdbc3: librerías que se utilizan para realizar las conexiones con los servidores de BD mysql y postgres.
- El Subsistema Weka: contiene los paquetes y clases necesarios para que los métodos de clusters y clasificaciones funcionen de forma correcta. Entre los paquetes y clases más importantes que se emplean se encuentran: los paquetes clusterers, classifiers, core y gui que agrupan las clases: Cobweb, X-Means y SimpleKMeans, J48, ArffViewer y Evaluation entre otras, que contienen los métodos útiles para el desarrollo del presente trabajo.

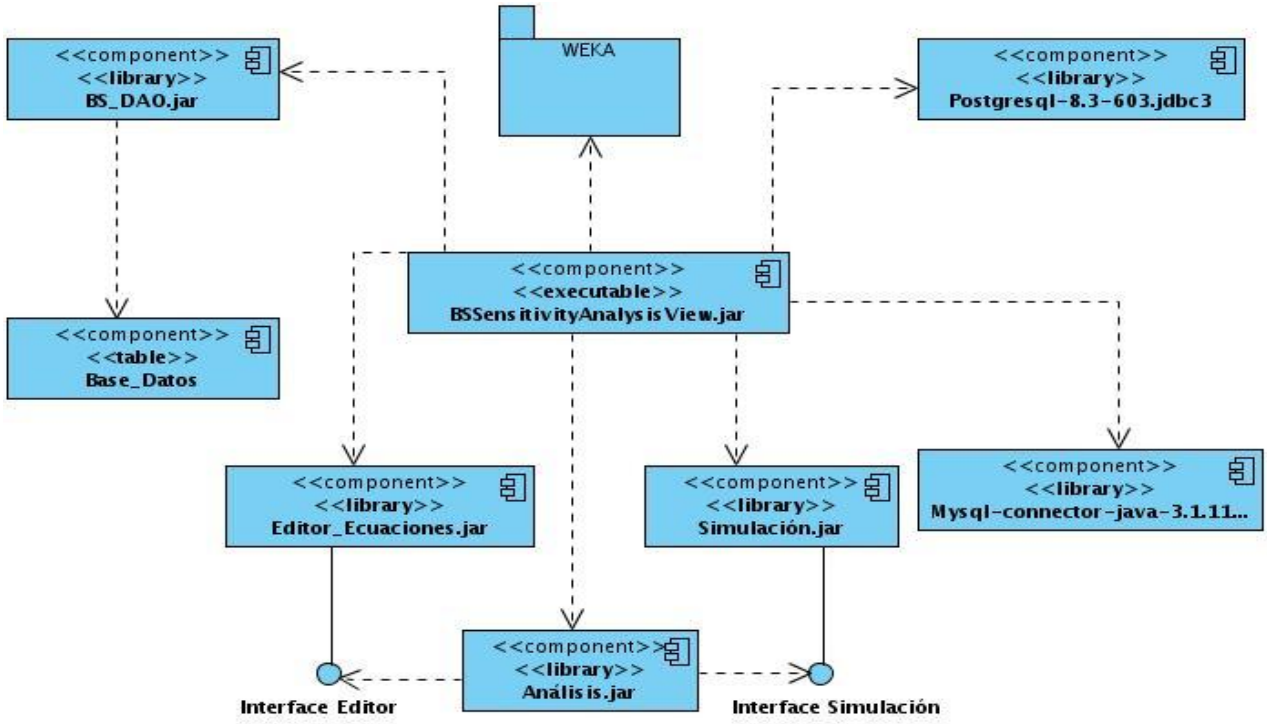

Figura 10: Diagrama de Componentes

3.3 Resultados obtenidos

Para obtener la aplicación deseada, se implementaron tres métodos principales:

- El método Comparar, para verificar la sensibilidad del parámetro.
- El método CompararMínimoCuadrado, para resolver la función objetivo.
- El método Sensitivity, para resolver el modelo matemático.

Además de estos métodos, se reutilizaron dos métodos de la versión 1.0 de BioSyS:

- El método Simular, para generar y asignar números aleatorios al modelo.
- El método Run, para levantar la interfaz principal de BioSyS y establecer la conexión a la base de datos.

Una vez implementado los métodos necesarios, la salida del sistema es visualizada gráficamente.

3.3.1 Método Comparar

Este método muestra como resultado una salida numérica que se obtiene de las simulaciones que están siendo analizadas. De esta forma el investigador puede ver el comportamiento de cada parámetro que se varió a lo largo del tiempo para tomar decisiones. Esta función ayuda a decidir cuál de los parámetros es sensible y cuál no.

En la figura 11, se muestra la implementación del método. El cual recibe como parámetro las simulaciones realizadas y compara con el porciento de error establecido por el usuario. La salida del método –como ya se explicaba anteriormente- es la sensibilidad del parámetro.

```
private double comparar(Simulacion simulacion, Simulacion simulacion0) {
    BSSet<Resultado> r1 = (BSSet<Resultado>) simulacion.getResultados();
    BSSet<Resultado> r2 = (BSSet<Resultado>) simulacion0.getResultados();

    double mayor = 0;
    double aux = 0;

    for (int i = 0; i < r1.size() && i < r2.size(); i++) {
        mayor = ((Resultado) r1.get(i)).getValor();
        aux = ((Resultado) r2.get(i)).getValor();

        if (mayor < aux) {
            mayor = aux;
        }
    }

    return mayor;

}
```

Figura 11: Método Comparar

3.3.2 Método CompararMínimoCuadrado

Este método es una variante del anterior, su objetivo es hallar la diferencia cuadrática, la cual no es más que el cuadrado de la diferencia entre el valor del vector de una simulación y su correspondiente en la otra. Para ello, el método busca el mínimo resultado de todos los que se han obtenido en las simulaciones que están siendo analizadas (ver figura 12)

```
private double compararMinimoCuadrado(Simulacion simulacion, Simulacion simulacion0){
    BSSet<Resultado> r1 = (BSSet<Resultado>) simulacion.getResultados();
    BSSet<Resultado> r2 = (BSSet<Resultado>) simulacion0.getResultados();

    double minimo = 0;

    for (int i = 0; i < r1.size() && i < r2.size(); i++) {
        double valor = ((Resultado) r1.get(i)).getValor() - ((Resultado) r2.get(i).getValor());
        minimo += Math.pow(valor, 2);
    }

    return minimo;
}
```

Figura 12: Método CompararMinimoCuadrado

3.3.3 Método SensitivityAnalysis

Este método utiliza el resultado de las simulaciones para generar un clasificador. Para ello se utiliza el algoritmo J48, que proviene de los árboles de decisión que están incluidos en el

Software Weka. Finalmente se muestra una gráfica con el resultado de la clasificación (ver figura 13).

```
private void sensitivityAnalysis(ArrayList<BSCondition> cond, ArrayList<BSCondition> |
    ArrayList<String> variables = new ArrayList<String>();
    ArrayList<String> parameters = new ArrayList<String>();
    File model = null;

    File arff = new File(BioSySControl.USER_BIOSYSDIRECTORY, "sensitivityAnalysis" + |
    arff.deleteOnExit();
    PrintStream arffFile = new PrintStream(arff);


    arffFile.print("@relation aa_clustered\n\n@attribute Instance_number numeric\n@at

    ArrayList<Integer> posicionesPar = new ArrayList<Integer>();
    ArrayList<Integer> posicionesVar = new ArrayList<Integer>();
    //ArrayList<Double> valores = new ArrayList<Double>();

    if (tool.equalsIgnoreCase("matlab")) {
        model = new File(BioSySControl.USER_BIOSYSDIRECTORY + File.separator + modelo
        model.deleteOnExit();
        BufferedWriter bw1 = new BufferedWriter(new FileWriter(model));
        bw1.write(modelo.getMatlabmodel());
        bw1.close();
        progress.setOutput(java.util.ResourceBundle.getBundle("simulation").getString
```

Figura 13: Método SensitivityAnalysis

3.3.4 Método Simular

El objetivo de este método es simular los datos necesarios para realizar la simulación del modelo matemático. Además, se encarga de ejecutar dicha simulación (Ver figura 14).

```
public void Simular() throws Exception {
    BSOdeSet odeSet = bsOdeSet.getOdeSet();
    odeSet.setNonNegative(bsParametersC.getNonNegativeVector());

    BSSensitivityAnalysis sensitivity = new BSSensitivityAnalysis(this, conn,
            modelo, odeSet, bsParametersC, bsOdeSet.getInitialTime(),
            bsOdeSet.getFinalTime(), bsOdeSet.getTool(), bsOdeSet.getType(),
            bsOdeSet.getInsert());
    sensitivity.start();
}
```

Figura 14: Método Simular

3.3.5 Método Run

La función de este método es crear y cerrar la conexión con la base de dato, así como, ejecutar el método simulación (Ver figura 15).

```
public void run() {
    try {
        bsDao = conn.getConnection();
        Simular();
        bsDao.cerrarConeccion();
    } catch (BSErrorMessage ex) {
        Logger.getLogger(BSSensitivityAnalysis.class.getName()).log(Level.SEVERE, null, ex);
        progress.Error(ex.getDetails());
        ErrorForm f = new ErrorForm(ex.getErrormessage(), ex.getDetails());
        f.show();
    } catch (Exception ex) {
        Logger.getLogger(BSSensitivityAnalysis.class.getName()).log(Level.SEVERE, null, ex);
        progress.Error();
        JOptionPane.showMessageDialog(null, ex.getMessage(),
java.util.ResourceBundle.getBundle("simulation").getString("ERROR"),
JOptionPane.ERROR_MESSAGE);
    } finally {
        if (matlab != null && matlab.isOpen()) {
            try {
                matlab.close();
            } catch (IOException ex) {
                Logger.getLogger(BSSensitivityAnalysis.class.getName()).log(Level.SEVERE, null,
ex);
            }
        }
        if (jopas != null && jopas.estaAbierto()) {
```

Figura 15: Método Run

3.3.6 Graficación de la Solución

Para graficar la solución se utilizó la herramienta Weka debido a la colección de algoritmos de aprendizaje automático de código abierto que posee, como son: J48, IBK (Instance Based Learning / Instancia Aprendizaje Basado en Problemas), BAGGING, K-Means, entre otros, los cuáles permiten aplicarlos directamente mediante una interfaz gráfica o ser llamados desde código Java.

Para la visualización de la sensibilidad de los parámetros, se va a utilizar los árboles de decisión y los clusters.

- **Árboles de decisión**

Para realizar este tipo de representación gráfica se utilizó el algoritmo j48 de WEKA, el cual se basa en la utilización del criterio ratio de ganancia (gain ratio). De esta manera se consigue evitar que las variables con mayor número de posibles valores salgan beneficiadas en la selección. Además el algoritmo incorpora una poda del árbol de clasificación una vez que éste ha sido inducido.

Entre sus principales características se destaca que:

1. Admite como variables predictoras atributos tanto numéricos como simbólicos (nominal), sin embargo requiere que la clase o variable a predecir sea de tipo nominal.
2. Permite el trabajo con pesos en los ejemplos. - Admite faltas en los atributos (valores perdidos) tanto en el entrenamiento como en la predicción del ejemplo. (36)

En la figura 16 se muestra un ejemplo de como quedaría visualizada la sensibilidad de los parámetros utilizando un árbol de decisión.

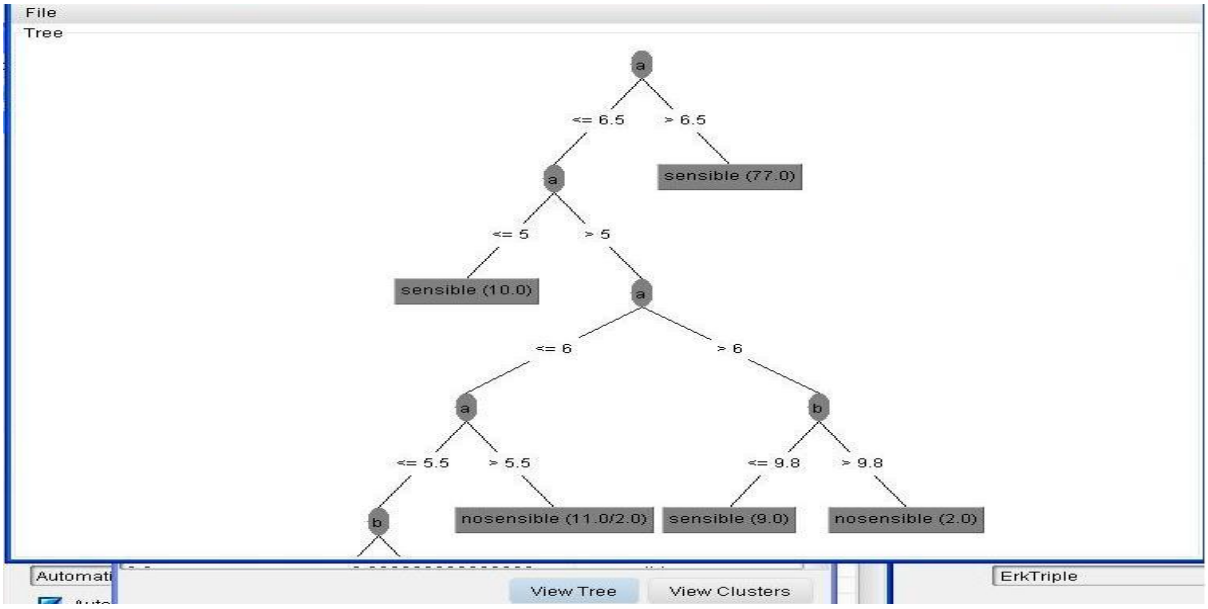

Figura 16: Árboles de Decisión

- **Clusters**

Para realizar la representación gráfica por clusters se utilizó el algoritmo K-Means. Básicamente este algoritmo busca formar clusters (grupos), los cuales serán representados por K objetos. Cada uno de estos K objetos es el valor medio de los objetos que pertenecen a dicho grupo. (37) En la figura 17 se muestra un ejemplo de como quedaría visualizada la sensibilidad de los parámetros utilizando Clusters.

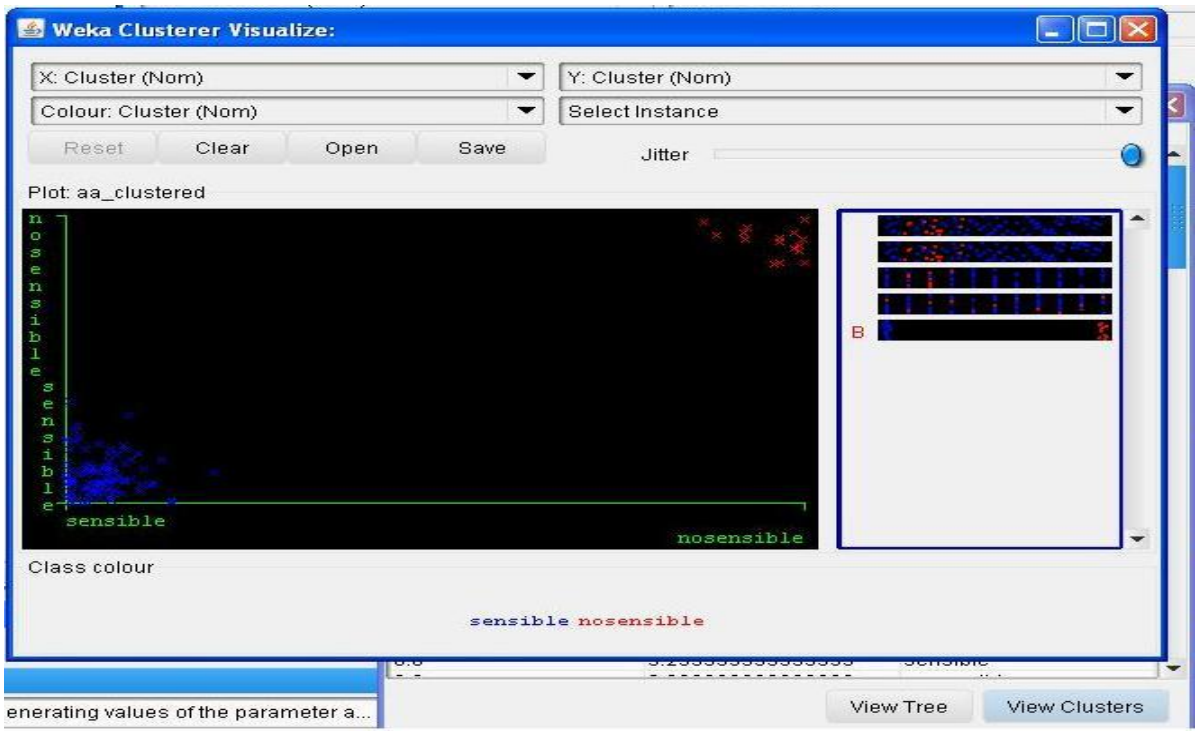

Figura 17: Clusters

3.4 Validación funcional

La calidad del software debe implementarse a lo largo de todo el ciclo de vida de un software, debe correr paralela desde la planificación del producto hasta la fase de producción de este como actividad de protección. El aspecto a considerar en el Control de la Calidad del Software es la “Prueba de Software”.

3.4.1 Pruebas de software

Las pruebas de software es un concepto que a menudo, es conocido como verificación y validación. Integra las técnicas de diseño de casos de prueba en una serie de pasos bien planificados que dan como resultado una correcta construcción del software. Entre algunas de las técnicas que se llevan a cabo para el proceso de prueba se encuentran las técnicas de caja negra y de caja blanca.

- **Prueba de caja negra**

Cuando se habla de un software, la prueba de caja negra se refiere a las pruebas que se llevan a cabo sobre la interfaz del mismo. Los métodos de prueba de la caja negra se centran en los requisitos funcionales del mismo e intentan encontrar errores de las siguientes categorías: 1) funciones incorrectas o ausentes; 2) errores de interfaz; 3) errores en estructuras de datos o en acceso a bases de datos externas; 4) errores de rendimiento y 5) errores de inicialización y terminación.

3.5 Validación funcional

Caso de Uso	Determinar Sensibilidad del Parámetro
Caso de Prueba	Obtener parámetros sensibles y no sensibles.
Entrada	Inicialmente el investigador introduce los parámetros a gestionar. Posteriormente escoge la función objetiva con que se va a obtener los resultados, introduce el índice de error y finalmente grafica.
Resultado esperado	Se obtiene una gráfica que muestra los resultados simulados.
Resultado de la prueba	Al introducir los datos de entrada y ejecutar el algoritmo de análisis de sensibilidad se obtiene que el resultado de la prueba fue el esperado, mostrando la gráfica con los resultados correctamente.

Tabla 5: Caso de prueba Determinar Sensibilidad de los Parámetros

En la figura 18 se muestra como se puede acceder al módulo de análisis de sensibilidad, el cuál está incluido en la plataforma BioSyS.

Análisis de sensibilidad

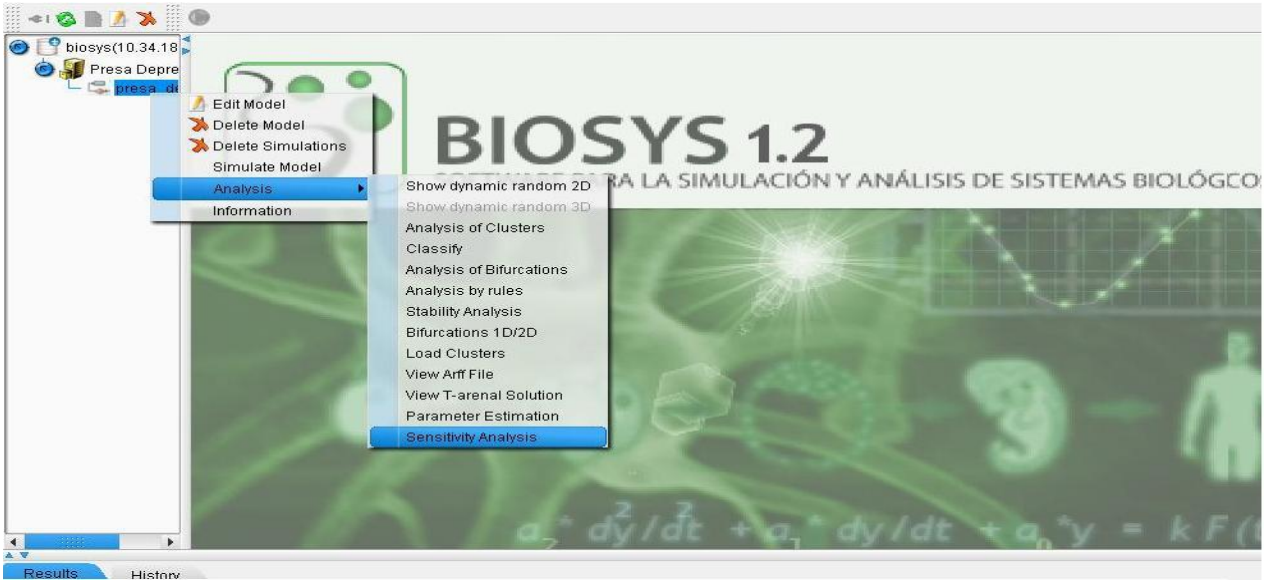

Figura 18: Acceso a Análisis de Sensibilidad

Entrada de Datos

En la figura 19 se muestra la ventana en la cual el usuario debe introducir los datos de entrada, para poder realizar el análisis de sensibilidad.

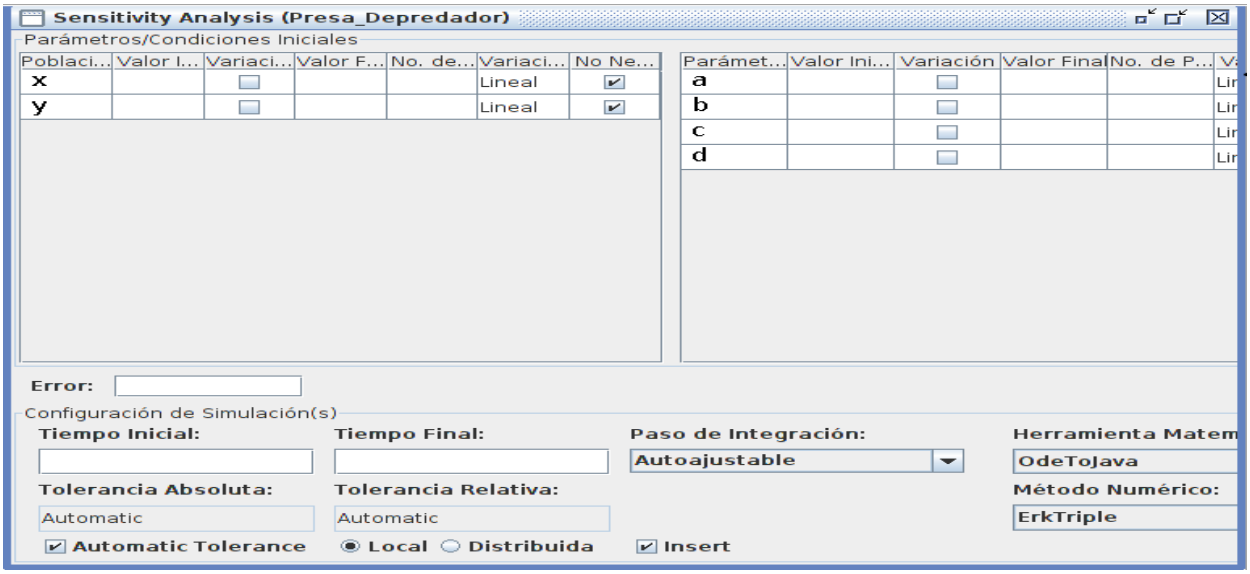

Figura 19: Datos de Entrada de "Determinar Sensibilidad del Parámetro"

Descripción de variables

Tabla 6: Descripción de variables

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
V1	-Vary	-Checkbox	NO	Opción obligatoria.
V2	Error	Campo de texto	NO	Sólo se admiten números, el campo es obligatorio.
V3	Initial Time	Campo de texto	NO	Sólo se admiten números, el campo es obligatorio.
V4	Final Time	- Campo de texto	NO	Sólo se admiten números, el campo es obligatorio.
V5	Parameters	-Celda de una Tabla	NO	Sólo se admiten números, el campo es obligatorio.
V6	Absolute Tolerance	- Campo de texto	NO	Sólo se admiten números, el campo es obligatorio.
V7	Relative Tolerance	- Campo de texto	NO	Sólo se admiten números, el campo es obligatorio.
V8	Automatic Tolerance	-Checkbox	SI	Opción aleatoria, selecciona la tolerancia si va a ser automática o no.

A continuación se presentan los casos de pruebas, para cada uno de los casos de uso planteados, a través de la matriz de datos, para comprobar que la aplicación funcione de forma correcta, donde:

V: indica válido, I: indica inválido, NA: que no es necesario proporcionar un valor del dato en este caso, ya que es irrelevante.

- V1, V2, Vn…Representan valores de entrada de datos.

Matriz de datos del caso de prueba: Determinar Sensibilidad de los Parámetros.

Escenario	Descripción	V1	V2	V3	V4	V5	V6	V7	V8	Respuesta del sistema	Flujo central
SC1.1. Determinar sensibilidad del parámetro con datos válidos.	En este escenario se determina la sensibilidad del parámetro. Comienza cuando el Investigador desea "Determinar Sensibilidad del Parámetro", el sistema evalúa la sensibilidad comparando dos distribuciones de valores del parámetro asociadas a los resultados de sensibles o no sensibles. El caso de uso termina una vez mostrado el resultado.	V	V double	V double	V double	V double	V double	V double	V	La salida será una tabla con los resultados obtenidos, donde se puede apreciar para que valores los parámetros que son sensible y para cuales no.	1. El Investigador selecciona realizar análisis de sensibilidad, el sistema muestra la interfaz. 2. El Investigador introduce los parámetros requeridos y el sistema valida los datos. 3. Selecciona el o los parámetros que va a variar, el sistema muestra una tabla con los valores obtenidos. 4. A partir de la tabla generada se pueden seleccionar las opciones: View Tree o View clusters. 5. El sistema muestra la gráfica.
SC1.1. Determinar sensibilidad de los parámetros con datos inválidos.			I Símbolos	I Símbolos	I Símbolos	I Símbolos	I Símbolos	I Símbolos		No se aceptan campos vacios, letras ni caracteres especiales.	
		I [ ]	I [ ]	I [ ]	I [ ]	I [ ]	I [ ]	I [ ]			
		V	I Símbolos	I	I	I	I	I		El sistemas muestra el mensaje de error: "Entre el tiempo inicial correctamente"	
		I [ ]	V	I Símbolos o [ ]	I Símbolos o [ ]	I Símbolos o [ ]	I Símbolos o [ ]	I Símbolos o [ ]		El sistemas muestra el mensaje de error: "Entre el tiempo inicial correctamente"	
		I [ ]	I Símbolos o [ ]	V	I Símbolos o [ ]	I Símbolos o [ ]	I Símbolos o [ ]	I Símbolos o [ ]		El sistemas muestra el mensaje de error: "Entre el tiempo final correctamente"	

		I [ ]	I Símbolos o [ ]	V	V	I Símbolos o [ ]	I Símbolos o [ ]	I Símbolos o [ ]		El sistemas muestra el siguiente mensaje de error: “Entre correctamente los valores de la condiciones iniciales”	
		I [ ]	I Símbolos o [ ]	I Símbolos o [ ]	I Símbolos o [ ]	V	I Símbolos o [ ]	I Símbolos o [ ]		El sistema no permite entrar datos incorrectos	
		V	V	V	V	V	V	I Símbolos o [ ]		El sistema muestra el siguiente mensaje de error: “Entre la tolerancia absoluta correctamente”	

Después de haber realizado las pruebas funcionales al caso de uso Determinar Sensibilidad de los Parámetros no se identificaron no conformidades, lo cual representa una buena implementación de caso de uso.

Interfaz del CU Determinar Sensibilidad de los Parámetros

En la figura 20 se muestra como se mostrarían los resultados una vez terminado el proceso de análisis de sensibilidad.

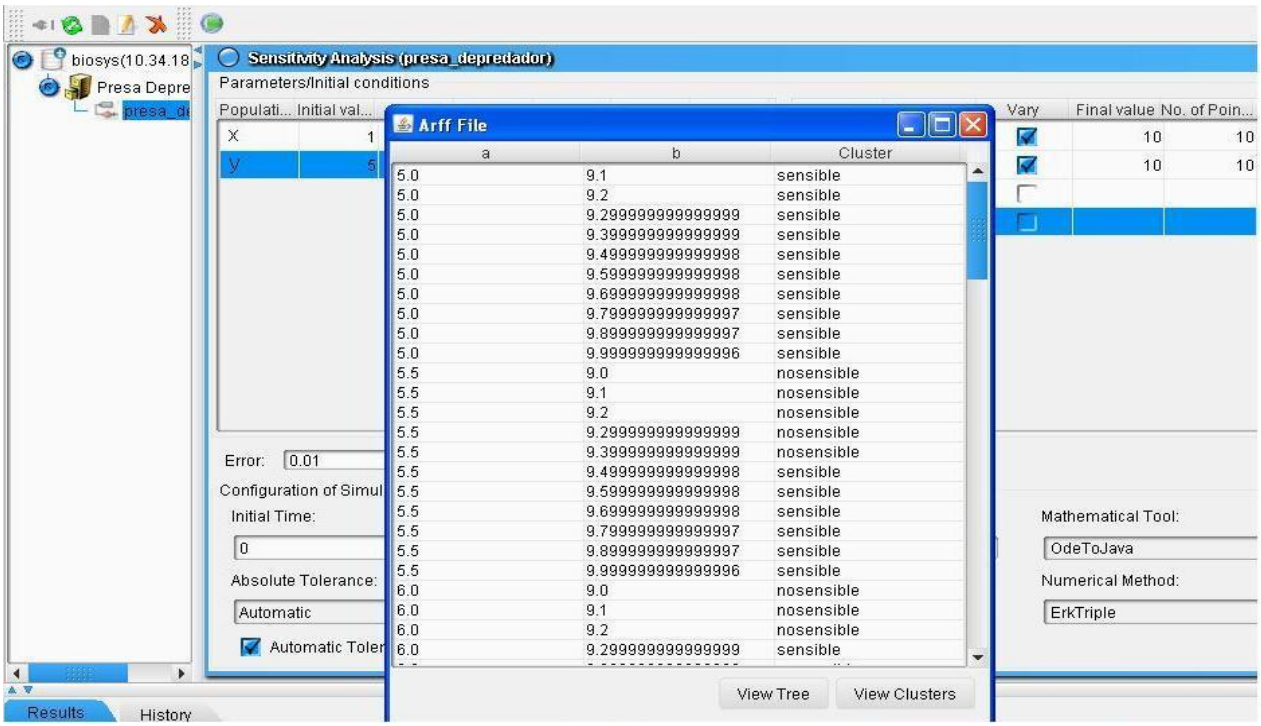

Figura 20: Resultado de "Determinar Sensibilidad de los Parámetros"

3.6 Conclusiones del capítulo

En el presente capítulo se realizó la implementación del sistema, para ello se llevó a cabo el diseño de los principales componentes y sus relaciones, mostrados de forma sencilla a través del Diagrama de Componentes. Además se hizo la modelación del diagrama de despliegue. Posteriormente se mostró el código de cinco métodos fundamentales para la aplicación. Para realizar la validación del sistema se definió usar la técnica de caja negra la cual fue aplicada en el diseño de pruebas, mostrando así las entradas y salidas válidas de la aplicación según los requerimientos.

Conclusiones

- Se realizó el estudio del arte sobre el análisis de sensibilidad, a partir de lo cual se definió utilizar la función de Mínimo Cuadrado para la realización de dicho análisis en la plataforma BioSyS.
- Se realizó el diseño de la aplicación basada en la metodología OpenUp.
- Se llevó a cabo la implementación del módulo de análisis de sensibilidad en la plataforma BioSyS.
- Se validó la solución propuesta mediante casos de prueba de caja negra, obteniéndose resultados satisfactorios.

Recomendaciones

- Realizar la implementación distribuida del Análisis de Sensibilidad, con el objetivo de optimizar el mismo.

Bibliografía

- **H. Christopher Frey, Ph.D.** Civli, Construction and Enviromental ENGINEERING. [En línea] http://www.ce.ncsu.edu/risk/pdf/frey.pdf.
- **Frenkel, Daan.** Forschungszentrum. [En línea] http://www2.fz-juelich.de/nic-series/volume23/frenkel.pdf.
- Scribd. [En línea] http://es.scribd.com/doc/37071710/Metodo-de-Monte-Carlo-I-M-sobol.
- **Meyer, Michael J.** Java Quant. [En línea] http://www.javaquant.net/books/MCBook-1.2.pdf.
- **Javier Faulín, Ángel A. Juan.** Universitat Oberta de Catalunya-UOC. [En línea] http://www.uoc.edu/in3/emath/docs/Simulacion_MC.pdf .
- **Kwang-Hyun Cho, Sung-Young Shin.** *Experimental Design in Systems Biology, Based on Parameter Sensitivity Analysis Using a Monte Carlo Method.*
- **Alberto Cardona, Norberto Nigro, Victorio Sonzogni, Mario Storti.** Asociación Argentina de Mecánica Computacional. [En línea] http://www.amcaonline.org.ar/ojs/index.php/mc/article/viewFile/517/492 .
- **M, Dr. Ing. Franco Bellini.** Investigación de Operaciones. [En línea] http://www.investigacion-operaciones.com/Curso_inv-Oper_carpeta/Clase12.pdf.
- Genoma España. [En línea] http://www.gen-es.org/12_publicaciones/docs/pub_76_d.pdf.
- **Mayorga, Luis S.** Facultad de Ciencias Médicas - UNCuyo. [En línea] http://revista.medicina.edu.ar/vol04_01/10/vol04_01_Art10.pdf.
- **Salinas, Germán Harvey Alférez.** Facultad de Ingeniería y Tecnoloía de la Universidad de Montemorelos. [En línea] http://fit.um.edu.mx/CIDET/publicaciones/COMP-004-2008%20Establecimiento%20de%20una%20Metodolog%C3%ADa%20de%20Desarrollo%20de%20Software%20para%20la%20Universidad%20de%20Navojoa%20Usando%20OpenUP.pdf.
- **López, Patricia.** OpenCourseWare . [En línea] http://ocw.unican.es/ensenanzas-tecnicas/ingenieria-del-software-i/practicas-1/is1-p01-trans.pdf.

- Webcache. [Online] [Cited: oct 15, 2010.] http://webcache.googleusercontent.com/search?q=cache:X9Z5cOh1ol4J:www.gestiopolis.com.
- **Kuo, Benjamin C.** Automatic control system. 1995.
- **Cho KH, Shin SY, Kolch W, Wolkenhauer O.** Experimental Design in System Biology on Parameter sensitivity Analysis Using a Monte carlo Method. 2003.
- **Submitted. Fassó, A., Cameletti, M., Bertaccini, P.** Unified Statistical Approach for Simulation, Modeling, Analysis and Mapping of Environmental Data . 2009.
- **DDEs. Wu WH, Wang FS, Chang MS.** Dynamic sensitivity analysis of biological systems. 2008.
- **Sobol, I. M.** Global Sensitivity Indices for Nonlinear Mathematical Models. 2004.
- **Mahdavi, Alborz and Davey, Ryan E.** Sensitivity analysis of intracellular signaling pathway kinetics predicts targets for stem cell fate control. 2007.
- Ecovirtual. [Online] [Cited: Nov 21, 2010.] http://ecovirtual.uncu.edu.ar/investigacion/2008/files/e_pdf/e_26_t_torre.pdf.
- Monografias. [Online] [Cited: oct 15, 2011.] http://www.monografias.com/trabajos12/moma/moma.shtml.
- **Escobar, Domingo.** Using petri nets to introduce operating system concepts. 1991.
- Dpto. de Ciencias de la Computación e Inteligencia Artificial. [Online] [Cited: Nov 12, 2010.] http://www.cs.us.es/cursos/ia2-2005/temas/tema-08.pdf.
- Laboratorio de Inteligencia Artificial-UTN-FRM. [Online] [Cited: Nov 10, 2010.] http://ai.frm.utn.edu.ar/micesari/files/RedesbayesianasMATILDE.pdf.
- PROYECTO ECUACIONES DIFERENCIALES . [Online] [Cited: Nov 10, 2010.] http://ecuacionesonline.blogspot.com/2010/10/sistema-de-ecuaciones-diferenciales.html.
- Uaq. [Online] [Cited: nov 12, 2010.] http://www.uaq.mx/matematicas/ed/programa.html.
- Berkeleymadonna. [Online] dic 5, 2010. http://www.berkeleymadonna.com.
- Klaus-schittkowski. [Online] [Cited: dic 5, 2010.] http://www.klaus-schittkowski.de.
- Math. [Online] [Cited: dic 11, 2010.] http://www.math.pitt.edu/~bard/xpp/xpp.html.

- Copasi. [Online] [Cited: dic 11, 2010.] http://www.copasi.org.
- Metodologías de desarrollo de software. [Online] [Cited: Febrero 13, 2011.] http://www.rhernando.net/modules/tutorials/doc/ing/met_soft.html.
- Scribd. [Online] [Cited: ene 19, 2011.] http://www.scribd.com/doc/2050925/metodologias-de-desarrollo-software.
- Cbasqa. [Online] [Cited: ene 26, 2011.] (http://cbasqa.wordpress.com/2008/09/02/proceso-de-desarrollo-openup/).
- Visual-paradigm. [Online] [Cited: ene 11, 2011.] http://www.visual-paradigm.com/.
- Netbeans. [Online] [Cited: Enero 8, 2011.] http://netbeans.org/index_es.html.
- Lenguajes-de-programacion. [Online] [Cited: ene 5, 2011.] http://www.lenguajes-de-programacion.com/lenguajes-de-programacion.shtml.
- Java. [Online] [Cited: ene 5, 2011.] http://java.ciberaula.com/articulo/que_es_java.
- Willydev. [Online] [Cited: ene 15, 2011.] http://www.willydev.net/descargas/prev/IntroArq.pdf.
- Egdamar. [Online] [Cited: feb 12, 2011.] http://egdamar877.blogspot.com/2009/05/expocicion.html..

Referencia bibliográfica

1. Webcache. [En línea] [Citado el: 15 de oct de 2010.] http://webcache.googleusercontent.com/search?q=cache:X9Z5cOh1ol4J:www.gestiopolis.com.

2. **Cho KH, Shin SY, Kolch W, Wolkenhauer O.** *Experimental Design in System Biology on Parameter sensitivity Analysis Using a Monte carlo Method.* 2003.

3. **Submitted. Fassó, A., Cameletti, M., Bertaccini, P.** *Unified Statistical Approach for Simulation, Modeling, Analysis and Mapping of Environmental Data .* 2009.

4. **DDEs. Wu WH, Wang FS, Chang MS.** *Dynamic sensitivity analysis of biological systems.* 2008.

5. **Sobol, I. M.** *Global Sensitivity Indices for Nonlinear Mathematical Models.* 2004.

6. **Mahdavi, Alborz y Davey, Ryan E.** *Sensitivity analysis of intracellular signaling pathway kinetics predicts targets for stem cell fate control.* 2007.

7. Ecovirtual. [En línea] [Citado el: 21 de Nov de 2010.] http://ecovirtual.uncu.edu.ar/investigacion/2008/files/e_pdf/e_26_t_torre.pdf.

8. Slideshare. [En línea] http://www.slideshare.net/jcoronelf/minimos-cuadrados-presentacion-final.

9. Laboratorio de Cómputo de la FIE. [En línea] http://lc.fie.umich.mx/~jrincon/apuntes%20intro%20min%20cuad.pdf.

10. Buenas Tareas. [En línea] [Citado el: 10 de Diciembre de 2011.] http://www.buenastareas.com/ensayos/Modelo-Matematico/1818941.html.

11. Monografias. [En línea] [Citado el: 15 de oct de 2011.] http://www.monografias.com/trabajos12/moma/moma.shtml.

12. **Escobar, Domingo.** *Using petri nets to introduce operating system concepts.* 1991.

13. Dpto. de Ciencias de la Computación e Inteligencia Artificial. [En línea] [Citado el: 12 de Nov de 2010.] http://www.cs.us.es/cursos/ia2-2005/temas/tema-08.pdf.

14. Laboratorio de Inteligencia Artificial-UTN-FRM. [En línea] [Citado el: 10 de Nov de 2010.]

http://ai.frm.utn.edu.ar/micesari/files/RedesbayesianasMATILDE.pdf.

15. PROYECTO ECUACIONES DIFERENCIALES . [En línea] [Citado el: 10 de Nov de 2010.] http://ecuacionesonline.blogspot.com/2010/10/sistema-de-ecuaciones-diferenciales.html.

16. Uaq. [En línea] [Citado el: 12 de nov de 2010.] http://www.uaq.mx/matematicas/ed/programa.html.

17. **Yang, Oh &.** 2000.

18. Berkeleymadonna. [En línea] 5 de dic de 2010. http://www.berkeleymadonna.com.

19. Klaus-schittkowski. [En línea] [Citado el: 5 de dic de 2010.] http://www.klaus-schittkowski.de.

20. Math. [En línea] [Citado el: 11 de dic de 2010.] http://www.math.pitt.edu/~bard/xpp/xpp.html.

21. Copasi. [En línea] [Citado el: 11 de dic de 2010.] http://www.copasi.org.

22. Metodologías de desarrollo de software. [En línea] [Citado el: 13 de Febrero de 2011.] http://www.rhernando.net/modules/tutorials/doc/ing/met_soft.html.

23. Scribd. [En línea] [Citado el: 19 de ene de 2011.] http://www.scribd.com/doc/2050925/metodologias-de-desarrollo-software.

24. Cbasqa. [En línea] [Citado el: 26 de ene de 2011.] (http://cbasqa.wordpress.com/2008/09/02/proceso-de-desarrollo-openup/).

25. Slideshare. [En línea] http://www.slideshare.net/guestf131a9/herramientas-case.

26. Visual-paradigm. [En línea] [Citado el: 11 de ene de 2011.] http://www.visual-paradigm.com/.

27. Lenguajes-de-programacion. [En línea] [Citado el: 5 de ene de 2011.] http://www.lenguajes-de-programacion.com/lenguajes-de-programacion.shtml.

28. Netbeans. [En línea] [Citado el: 8 de Enero de 2011.] http://netbeans.org/index_es.html.

29. Entorno Virtual de Aprendizaje. [En línea] 2011. http://eva.uci.cu.

30. Willydev. [En línea] [Citado el: 15 de ene de 2011.] http://www.willydev.net/descargas/prev/IntroArq.pdf.

31. Egdamar. [En línea] [Citado el: 12 de feb de 2011.]

http://egdamar877.blogspot.com/2009/05/expocicion.html.

32. **Larman, Craig.** *UML y Patrones.* s.l. : Indiana, 2004.

33. Sparxsystems. [En línea] [Citado el: 15 de febrero de 2011.] http://www.sparxsystems.com.ar/download/ayuda/index.html?deploymentdiagram.htm.

34. Entorno Virtual de Aprendizaje. [En línea] [Citado el: 15 de Febrero de 2011.] http://eva.uci.cu/file.php/259/Curso_2010-2011/Semana_1/Conferencia_1/Materiales_basicos/Diseno.pdf.

35. Departamento de Sistemas Informáticos. [En línea] [Citado el: 18 de Febrero de 2011.] http://www.dsi.uclm.es/asignaturas/42530/pdf/M2tema12.pdf.

36. **Fernández, Gretel.** The Distributed Group. [En línea] http://www.tdg-seville.info/Download.ashx?id=48.

37. **FLEMING, R. J. S:Uso de la simulación computarizada para avanzar la investigación agrícola en los países en vía de desarrollo Estados Unidos de Norteamérica.** lastate. [En línea] 8 de Junio de 2011. http://www.public.iastate.edu/~rjsalvad/manuscripts/general/simulacion.html.

38. **A. Saltelli, K. Chan and M. Scott, eds.** *Sensitivity Analysis.* 2009 .

39. **Kuo, Benjamin C.** *Automatic control system.* 1995.

40. Java. [En línea] [Citado el: 5 de ene de 2011.] http://java.ciberaula.com/articulo/que_es_java.

41. Catarina. [En línea] [Citado el: 20 de Mayo de 2011.] http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/moreno_a_jl/capitulo5.pdf.

42. **Fernández, Gretel.** The Distributed Group. [En línea] http://www.tdg-seville.info/Download.ashx?id=48.

Anexos

Anexo 1 Diagramas de Secuencia

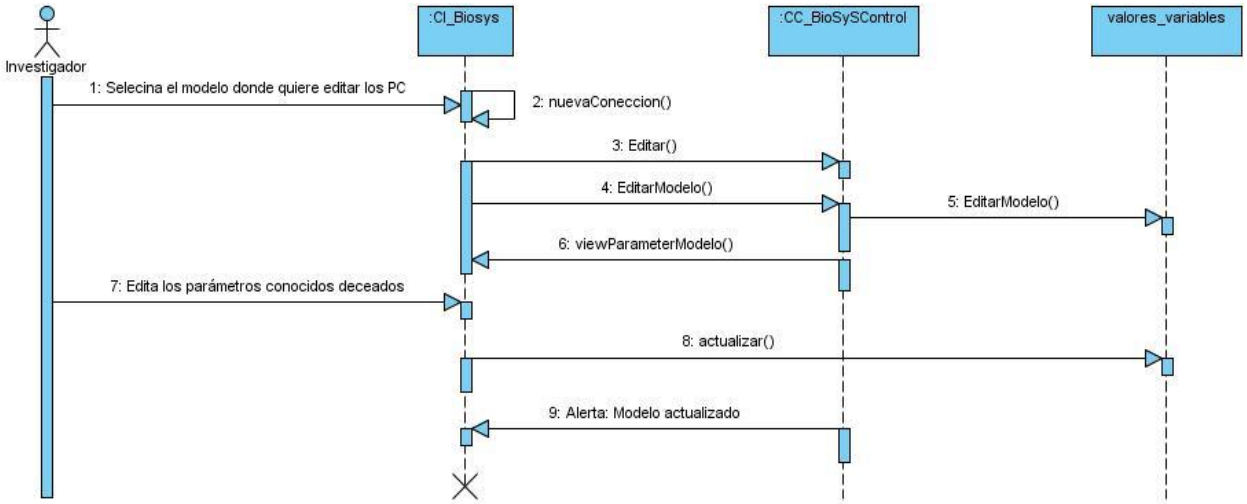

Figura 21: Editar Parámetros Conocidos

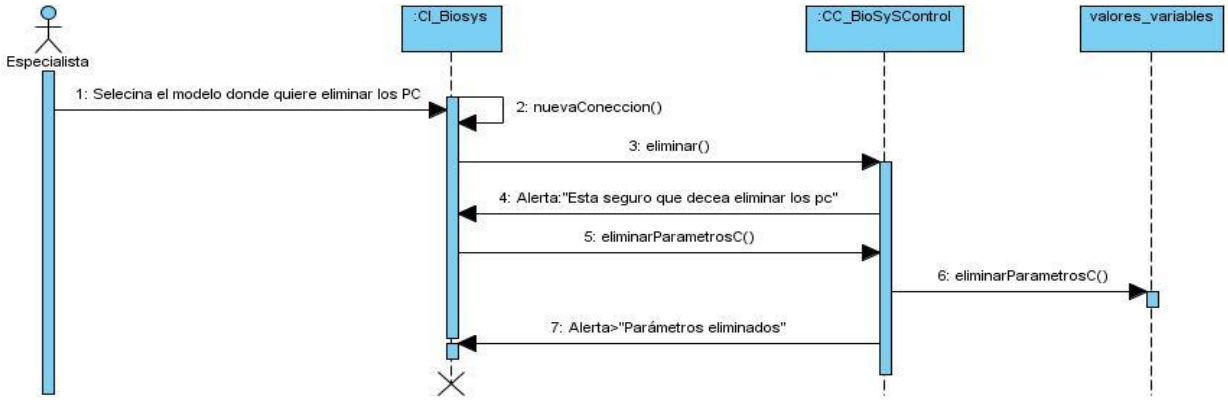

Figura 22: Eliminar Parámetros Conocidos

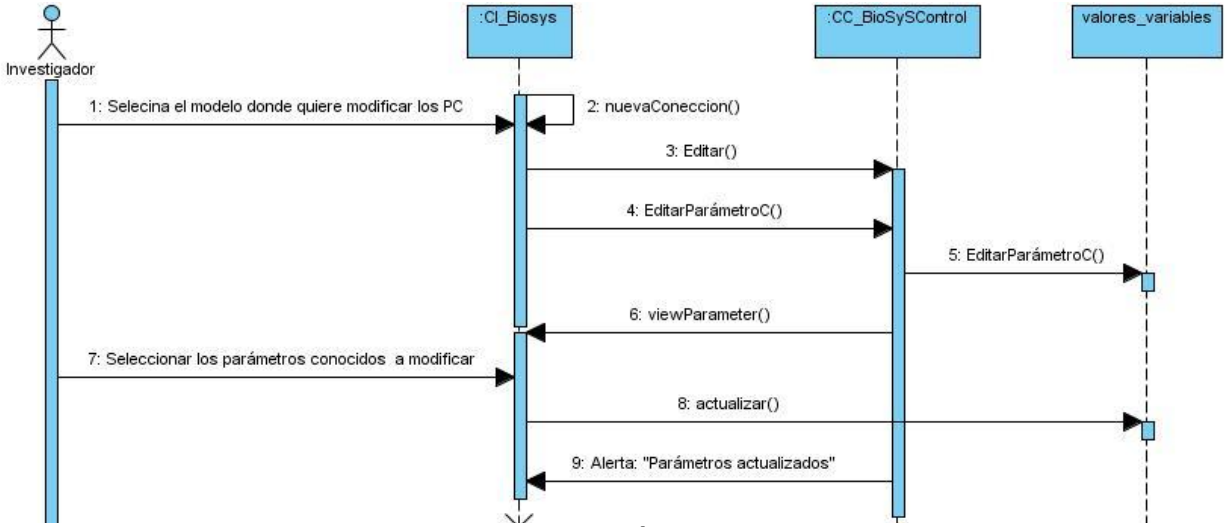

Figura 23: Modificar Parámetros Conocidos

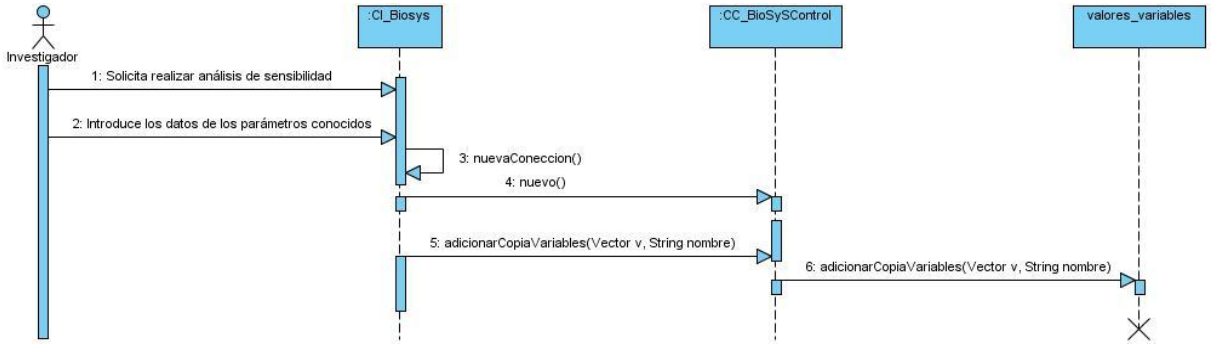

Figura 24: Insertar Parámetros Conocidos

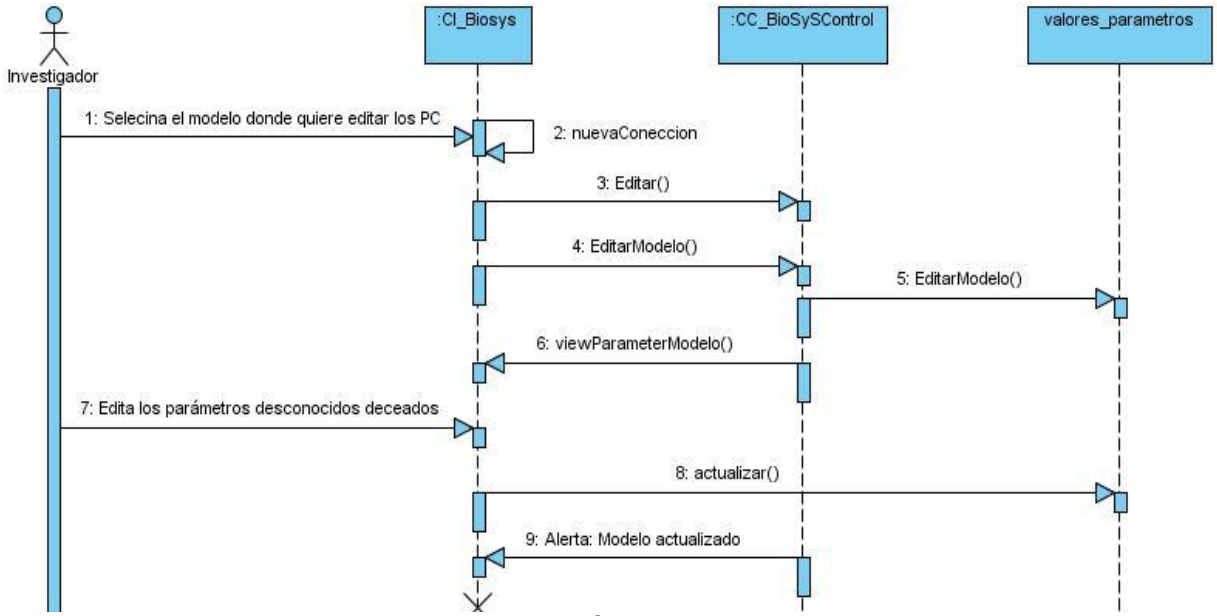

Figura 25: Editar Parámetros Desconocidos

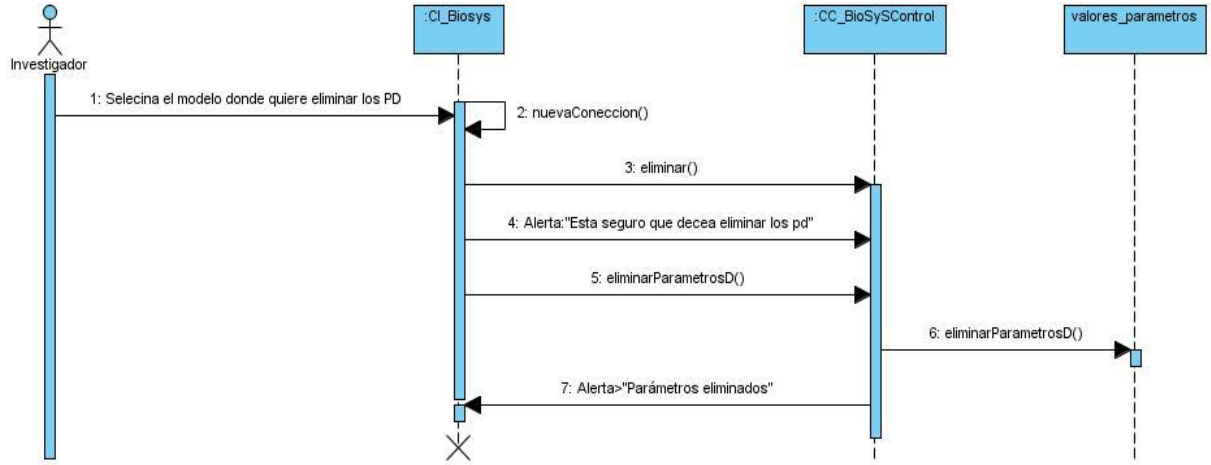

Figura 26: Eliminar Parámetro Desconocido

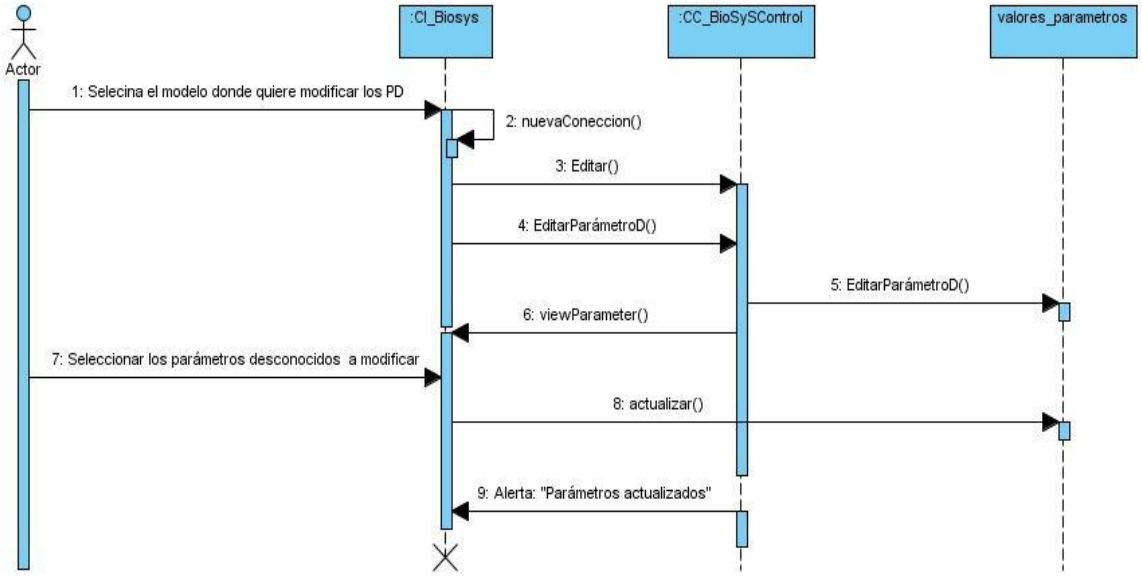

Figura 27: Modificar Parámetros desconocidos

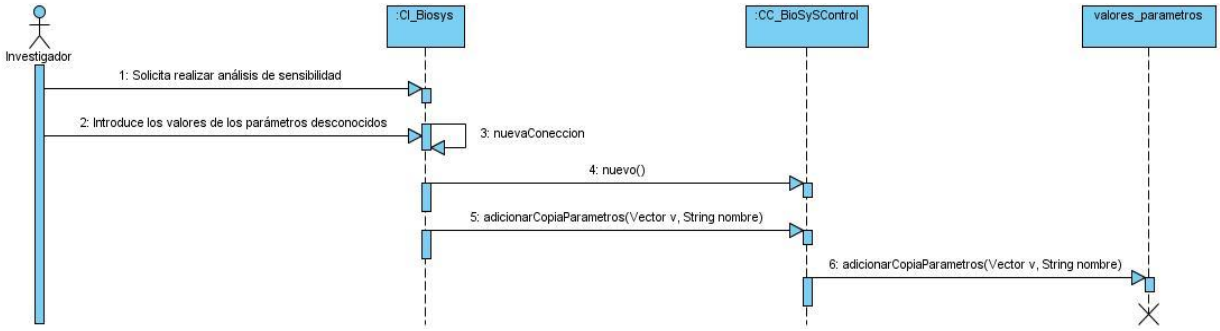

Figura 28: Insertar Parámetros Desconocidos

Glosario

- **Bioinformática:** Es la aplicación de los ordenadores y los métodos informáticos en el análisis de datos experimentales y simulación de los sistemas biológicos. Las principales aplicaciones de la Bioinformática son la simulación, la minería de datos y el análisis de los datos obtenidos en un estudio.
- **Clustering:** Se basa en intentar responder cómo es que ciertos objetos (casos) pertenecen o "caen" naturalmente en cierto número de clases o grupos, de tal manera que estos objetos comparten ciertas características.
- **Dicotómicos: dico** se refiere a dos y **tómico** a partes, es decir se refiere a la división en 2 partes. Puede tratarse de la bifurcación de un tallo o de una rama.
- **Distribución normal:** en estadística y probabilidad se llama, distribución normal, a una de las distribuciones de probabilidad de variable continua que con más frecuencia aparece en fenómenos reales.
- **Función discriminante lineal:** es de un polinomio que permite determinar los diferentes tipos de soluciones que existen en una ecuación. (si hay soluciones reales, si hay soluciones repetidas, o si hay soluciones complejas).
- **Integrales Multidimensionales:** contienen funciones hipergeométricas y exponenciales complejas, lo que les confieren un carácter altamente oscilatorio.
- **Metodología:** Define quién hace qué, cómo y cuándo.
- **Modelo:** Representación abstracta de la realidad.
- **Patrones:** Unidad de información nombrada, instructiva e intuitiva que captura la esencia de una familia exitosa de soluciones probadas a un problema recurrente dentro de un cierto contexto.
- **Requisitos:** Capacidades o condiciones que se deben cumplir.
- **Sistema:** Conjunto de componentes que interaccionan entre sí para lograr un objetivo común.
- **Software:** Término genérico que designa al conjunto de programas que posibilitan realizar una tarea específica en un ordenador.
- **Verosimilitud:** probabilidad.