Universidad de las Ciencias Informáticas.

Facultad 6.

Software para la Simulación de Sistemas Biológicos.

Módulo “Editor de Ecuaciones Diferenciales”.

Trabajo de Diploma para optar por el título de Ingeniero Informático

Autores: Ileana Centelles Rubiera.

Jose Luis Cespedes Martínez.

Tutores: Lic. Noel Moreno Lemus.

Lic. Gilberto Arias Naranjo.

Ing. Anthony R: Sotolongo Leon.

Co-tutor: Lic. Eduardo Hermenegildo Caballero Santana.

Consultante: Dr. Kalet León Monzón.

La Habana, 2007.

“Año 49 de la Revolución”

DECLARACIÓN DE AUTORÍA

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

Para que así conste firmo la presente a los _***3***_ días del mes de __**Julio**__ del año __**2007**__.

Lic. Noel Moreno Lemus.

Lic. Gilberto Arias Naranjo.

Ing. Anthony R: Sotolongo Leon.

Lic. Eduardo Hermenegildo Caballero Santana.

_____ Firma del Autor

_____ Firma del Autor

_____ Firma del Tutor

_____ Firma del Tutor

_____ Firma del Tutor

_____ Firma del Tutor

DATOS DE CONTACTO

Tutores:

Lic. Noel Moreno Lemus.
Universidad de las Ciencias Informáticas, Habana, Cuba.
Email: noel@uci.cu

Lic. Gilberto Arias Naranjo.
Universidad de las Ciencias Informáticas, Habana, Cuba.
Email: gilbertoa@uci.cu

Ing. Anthony R. Sotolongo León.
Universidad de las Ciencias Informáticas, Habana, Cuba.
Email: asotolongo@uci.cu

Co-tutor:

Lic. Eduardo Hermenegildo Caballero.
Universidad de las Ciencias Informáticas, Habana, Cuba.
Email: ecaba@uci.cu

Consultante:

Dr. Kalet León Monzón.
Centro de Inmunología Molecular, Habana, Cuba.
Email: kalet@ict.cim.sld.cu

"El hombre puede hacer de si mismo muchas cosas producto de su propio esfuerzo físico y espiritual, el que se proponga cultivar la virtud la cultiva, el que se proponga alcanzar los más altos niveles de conocimiento los alcanza."

Fidel Castro Ruz.

Agradecimientos

Agradecemos a la Revolución por brindarnos la posibilidad de formarnos como profesionales y habernos hecho partícipes de este gran proyecto de nuestro Comandante en Jefe. A los tutores por el apoyo brindado, a los profesores por nuestra formación, a nuestros familiares y amigos, a todos aquellos que han contribuido a la realización de este trabajo y han compartido junto a nosotros las tensiones y a los que en un momento determinado nos preguntaron ¿Y la Tesis?

A todos, de corazón

Gracias.

.

Dedicatoria:

A nuestros padres.

A nuestros abuelos y hermanos.

A nuestra familia toda.

Resumen:

La modelación matemática es una de las herramientas que se utilizan hoy en día para el estudio de problemas en medicina y biología, fisiología, bioquímica, farmacocinética; sus objetivos primordiales son describir, explicar y predecir fenómenos y procesos en dichas áreas. Los modelos matemáticos nos permiten representar un problema médico o biológico de una manera objetiva en la que se define una serie de relaciones matemáticas entre las mediciones cuantitativas del problema y sus propiedades. Sin embargo, para ser utilizado como método de experimentación y establecer hipótesis, no existe en la actualidad una herramienta informática que le permita a los investigadores relacionados con la materia: registrar, editar, gestionar, comprobar y analizar los sistemas modelados como apoyo a las herramientas de modelación gráfica de sistemas biológicos en un mismo entorno de trabajo.

La investigación tiene como objetivo desarrollar el análisis y diseño de una herramienta informática que brinde todas las posibilidades básicas para el trabajo con las ecuaciones matemáticas, dar formato y leer ficheros MathML, que cuente con una biblioteca con funciones predefinidas e interactúe con herramientas de análisis matemático en un mismo entorno de trabajo, creando así, una herramienta integrada para el estudio de los modelos matemáticos.

Índice

Introducción

En los últimos 10 años, se han aplicado tecnologías genómicas y proteómicas en la identificación y desarrollo de una generación de tratamientos para diversas enfermedades. Mientras estas tecnologías se han automatizado cada vez más, produciendo una avalancha de posibles objetivos terapéuticos y novedades biológicas, los cálculos estiman que los costos y el tiempo de desarrollo de medicamentos por separado seguirán creciendo y que pronto excederá al billón de dólares. Una importante razón para este aumento de costo es el enorme número de compuestos que fracasan clínicamente tras varios años de desarrollo pre-clínico (alrededor del 50% de los compuestos fracasan durante los primeros ensayos clínicos). Es necesario por tanto un contexto que integre las interacciones dinámicas de los mecanismos reguladores que distinguen la salud de la enfermedad y que, en consecuencia, prediga los resultados terapéuticos. La biosimulación, también denominada “modelling” por ordenador o biología in silico, es una solución de la biología de sistemas para este problema, que utiliza la abundancia de diferentes datos para construir un modelo dinámico de fisiología humana.

En este campo se han obtenido grandes avances tales como:

- Desarrollo de técnicas experimentales que miden con exactitud muchas cantidades y flujos biológicos.
- Avance en la comprensión y reproducción matemática de sistemas robustos y complejos.
- Mejoras en la capacidad informática que permiten la simulación de sistemas grandes y complejos en económicos ordenadores personales.

Incluso con estos avances, la complejidad típica que se necesita para las biosimulaciones a gran escala hace que desarrollarlas resulte un desafío y se necesiten amplios recursos, entre otros un software sofisticado de reproducción y análisis. La aplicación de esos modelos en el descubrimiento de medicamentos y en el proceso de desarrollo implica varios desafíos adicionales, como:

- Crear un software que permita que estos modelos se desarrollen, se modifiquen, se expandan y se confirmen con rapidez a la vez que lo hace el estado actual de los conocimientos.
- Modificar la organización de los departamentos de investigación de modo que se facilite la formación de equipos multidisciplinarios.

- Formar a matemáticos, ingenieros y científicos para integrar un equipo multidisciplinario que desarrolle complejos modelos de enfermedad y dirijan las investigaciones in silico.

Durante los últimos años ha habido un reconocimiento de estos desafíos, y se están haciendo distintos esfuerzos por superarlos. Muchas universidades están añadiendo departamentos de bioingeniería y se han creado varios institutos de biología de sistemas en Asia, América y Europa. Así mismo existen compañías y grupos de investigación académica que están desarrollando modelos matemáticos y software diseñados específicamente para estudiar complejos sistemas biológicos, dentro de las cuales encontramos:

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

Todas estas empresas parten de la propiedad intelectual sobre una plataforma de modelación (software) para integrar información, representar matemáticamente y simular sistemas biológicos. Muchos de los cuales se basan principalmente en la modelación gráfica de dichos modelos por el alto grado de expresividad y comprensión que ofrece, a diferencia de la modelación matemática que lleva un mayor grado de complejidad. La modelación gráfica genera un modelo matemático con el que posteriormente se realizan las simulaciones dados los valores que se definen en los parámetros a evaluar. Muchas veces los investigadores necesitan realizar cambios en la estructura de estos modelos matemáticos para el estudio del sistema en cuestión, necesitando comprobar la consistencia del nuevo sistema de ecuaciones creado, así como reducir el número de parámetros libres a evaluar, para ello se auxilian del empleo de herramientas de análisis matemáticos. Estas herramientas requieren un dominio de las mismas por parte del investigador, además no permiten mantener un registro de las transformaciones realizadas a estos modelos, por lo que los investigadores se ven forzados a repetir las acciones para su introducción cada

vez que necesiten realizar un análisis o transformación sobre un mismo modelo, es decir carecen de un entorno homogéneo que facilite el trabajo con los modelos matemáticos en general.

Dentro del proyecto de informatización del Sistema de Salud Nacional y los Centros del Polo Científico, la Universidad de Ciencias Informáticas conjuntamente con el Centro de Inmunología Molecular (CIM), se encuentra desarrollando un software que permita la integración de todas las funcionalidades básicas para la biosimulación con el fin de facilitar la obtención y producción de nuevos biofármacos destinados al tratamiento del cáncer y otras enfermedades crónicas no trasmisibles. Se hace necesario incluir a dicho software un componente que sirva como puente entre la modelación gráfica y la simulación y análisis de los resultados de los modelos generados, que permita realizar un conjunto de transformaciones a los modelos y validarlos desde el punto de vista simbólico para comprobar que el referido sistema sea coherente así como ajustarlo al estudio que dentro de los mismos se pretende realizar mediante la interacción con herramientas matemáticas un mismo entorno.

Por ello se plantea el siguiente ***Problema Científico***:

¿Cómo unificar los procesos de cálculo simbólico y edición de modelos matemáticos?

Objeto de estudio

Procesos de edición y análisis de modelos matemáticos.

Campo de acción.

Procesos de edición y análisis de modelos matemáticos conformados por ecuaciones algebraicas y sistemas de ecuaciones diferenciales.

Objetivos del trabajo

Objetivo general

Analizar y diseñar una herramienta computacional para la integración de los procesos de edición y cálculo simbólico de modelos matemáticos conformados por Sistemas de Ecuaciones Diferenciales.

Objetivos específicos

- Modelar el negocio.
- Levantar los requerimientos del sistema.
- Analizar y diseñar una herramienta computacional para el trabajo con expresiones matemáticas.

Tareas desarrolladas para cumplir los objetivos.

- Realización de entrevistas al cliente.
- Estudio de Editores de ecuaciones existentes.
- Estudio y selección de la metodología y herramientas para el análisis y diseño del software.
- Selección de los artefactos a utilizar según la metodología seleccionada.
- Estudio del estándar MathML para el trabajo con la información matemática.
- Estudio del lenguaje de programación Java.
- Estudio de métodos matemáticos para realizar el cálculo simbólico.
- Realización del análisis y diseño de una herramienta computacional para la integración de los procesos de de edición y cálculo simbólico de modelos matemáticos conformados por Sistemas de Ecuaciones Diferenciales.

Estructura del Documento.

El documento está estructurado en capítulos donde se van describiendo los pasos que se desarrollan con vista alcanzar los Objetivos del trabajo.

Capítulo 1: Fundamentación Teórica.

Este capítulo comprende el estado del arte del tema tratado, hace una breve explicación del problema, las tendencias actuales que existen con la edición de las matemáticas y su estudio mediante herramientas computacionales. Se describe la justificación de las tecnologías y metodologías utilizadas para la solución del problema.

Capítulo 2: Modelo de Dominio.

En este capítulo se describe los principales conceptos y entidades que existen en el Dominio. Se representan en un modelo las interacciones entre las entidades involucradas en el dominio.

Capítulo 3: Requerimientos del Sistema.

En este capítulo se describe los Requisitos Funcionales y No Funcionales del sistema, así como los Actores que intervienen en el mismo, también se encuentra el Diagrama de Casos de Uso y la descripción textual de cada uno de ellos.

Capítulo 4: Diseño del Sistema.

En este capítulo se confecciona el Modelo del Diseño del Sistema, definiéndose cada una de las clases que compondrán al mismo y se realizan los diferentes diagramas de interacción presentes entre dichas clases.

Capítulo 1: Fundamentación Teórica.

Introducción.

En este capítulo se describe la fundamentación teórica Se verán temas relacionados con la Modelación Matemática, en especial la modelación vinculada a sistemas biológicos, resaltándose la importancia de la misma, así como una descripción de los análisis fundamentales para la comprobación de la consistencia de los mismos. Se hace una explicación de la utilización de las matemáticas por los investigadores y cómo se codifican actualmente a través del estándar MathML al igual que se describen algunos editores de ecuaciones matemáticas existentes y se explican algunas de las principales tendencias y tecnologías actuales que se tienen en cuenta durante el desarrollo del presente trabajo.

1.1 Modelación Matemática.

La biología está llena de problemas muy complicados, y ha continuado avanzado muy rápidamente en los últimos años. Eso está haciendo que los biólogos tengan un montón de datos, tantos que no saben qué hacer con ellos. De ahí la importancia de las matemáticas al adquirir un gran valor a la hora de buscar un sentido al flujo cada vez mayor de datos biológicos.

La modelación matemática es una de las herramientas que se utilizan hoy en día para el estudio de problemas en medicina, biología, fisiología, bioquímica y farmacocinética. Sus objetivos primordiales son describir, explicar y predecir fenómenos y procesos en dichas áreas; permitiendo representar un problema médico o biológico de una manera objetiva en que se definen una serie de relaciones matemáticas entre las mediciones cuantitativas del problema y sus propiedades. Asimismo puede existir infinidad de modelos para infinidad de problemas biológicos, al igual que existen diferentes modelos para un mismo problema, pues, como norma, todo modelo matemático que arroje resultados acordes con los obtenidos experimentalmente debe ser aceptado, además un mismo modelo es aplicable a diferentes procesos o fenómenos.

Dos de los modelos más utilizados con este fin, son los Autómatas Celulares y la Dinámica de Poblaciones.

1.1.1 Modelación de Sistemas Biológicos por Dinámica de Poblaciones.

Los sistemas biológicos que se pueden analizar desde la óptica de la dinámica de poblaciones son descritos de forma general por la Ley de Malthus; que no es más que una ecuación diferencial con condiciones iniciales o Problema de Cauchy.

$$dP/dt = k * P, P(0) = P_0$$

donde:

P: población (cantidad de miembros de la población).

t: tiempo (tiempo de estudio de la población).

k: Es una función f(P_1,..., P_n, t), donde P_1 ,...,P_n representan las poblaciones presentes en el modelo. Es muy común que estas funciones sean definidas como constantes, f(P_1,..., P_n, t)= a, a ϵ R.

La solución para este problema de Cauchy es:

$$P = P_0 * e^{k*t}$$

Estos modelos representan poblaciones y compuestos químicos que interactúan mutuamente mediante diferentes procesos dentro de los que se encuentran: la proliferación y muerte de una población, diferenciación o tránsito de una población a otra, la producción de compuestos químicos que puedan generarse como resultado del metabolismo de una población y la acción de poblaciones y sustancias químicas como reguladores de la velocidad de un proceso actuando como catalizadores o inhibidores.

La variación en el tiempo de la cantidad de población y de sustancia, respectivamente, es descrita por ecuaciones diferenciales que son originadas por cada uno de los procesos en que toma parte la población o el compuesto químico en cuestión.

Cada ecuación que describa el comportamiento de una población tiene que tener en cuenta todos los procesos en que participa dicha población; esto se logra expresando la ecuación como una sumatoria donde cada término representa la acción de un proceso en la población en cuestión. [3]

1.1.2 Análisis de la Consistencia del Modelo.

Cada paso en la formulación de un modelo debe ser corroborado, con lo cual se valida el mismo, en consecuencia, el modelo debe ser consistente, para ello se comprueba si cumple con el Principio de Conservación de la Masa y si es dimensionalmente correcto.

Una vez confirmada la consistencia de un modelo matemático, es posible pasar a la fase de aplicación que, sin lugar a dudas, es la más importante. Entonces es cuando el modelo se usa como herramienta para pronosticar resultados experimentales que tal vez sean demasiado difíciles de abordar a través de estudios clínicos.

Los resultados llevan a conjeturas e hipótesis que posiblemente puedan corroborarse de otra forma y así avanzar el entendimiento del mundo en que vivimos.

1.1.3 Análisis Conservación de la Masa

Este tipo de modelo es conocido también como modelos compartamentales los cuales hacen uso de ecuaciones de balance de masas para representar cada comportamiento. Dichas ecuaciones siguen el principio de Conservación de la Materia esta vez aplicado a la modelación de los sistemas biológicos. Este análisis se basa en la comprobación de que la cantidad neta entrante debe ser igual a la cantidad neta saliente, de manera que la suma de ambas sea igual a cero. Dicho principio no se cumplirá para procesos de proliferación o muerte de una población ya que surgen y se pierden elementos.

1.1.4 Análisis Dimensional.

Con el análisis dimensional se puede establecer si un conjunto de ecuaciones son correctas o no, e incluso saber si estamos escalando adecuadamente un sistema o proceso. El Análisis Dimensional se basa en que las ecuaciones deben ser homogéneas, es decir, las dimensiones de las magnitudes a ambos lados de una igualdad deben ser idénticas.

El método a seguir consiste en cambiar el conjunto original de parámetros de entrada dimensionales involucradas en un problema por otro conjunto de parámetros de entrada adimensionales más reducido. Estos parámetros adimensionales se obtienen mediante combinaciones adecuadas de los parámetros

dimensionales y no son únicos, aunque sí lo es el número mínimo necesario para estudiar cada sistema. [10]

De este modo, al obtener uno de estos conjuntos de tamaño mínimo se consigue:

- Analizar con mayor facilidad el sistema objeto de estudio.
- Reducir drásticamente el número de ensayos que debe realizarse para averiguar el comportamiento o respuesta del sistema.

Es posible por tanto expresar una dimensión dependiente en función de un conjunto seleccionado de dimensiones básicas independientes, en este caso se utiliza el Sistema Internacional de unidades, estas dimensiones básicas son:

- L, longitud.
- M, masa.
- T, tiempo.
- K, grados kelvin.

Las que serán tratadas como grupos adimensionales cuya dimensión es 1.

De manera que las dimensiones de área, volumen, velocidad y aceleración se representarían tal y como se muestran en la siguiente tabla.

Sistema	Área (L^2)	Volumen(L^3)	Velocidad(L/T)	Aceleración(L/T^3)
SI	m^2	m^3	m/s	m/s^2
cgs	cm^2	cm^3	cm/s	cm/s^2
De ingeniería británico	pie^2	pie^3	pie/s	pie/s^2

Tabla 1 Ejemplo de expresión de las dimensiones con parámetros adimensionales.

El análisis dimensional aprovecha el hecho de que las dimensiones pueden tratarse como cantidades algebraicas. Es decir, las cantidades pueden sumarse o restarse sólo si se tienen las mismas dimensiones, por lo que los términos en ambos lados de una ecuación deben tener las mismas dimensiones.

Un ejemplo es el siguiente:

Hay que mostrar que la expresión v = vo + *at* es dimensionalmente correcta, donde v y vo representan velocidades, ***a*** es la aceleración y ***t*** es un intervalo de tiempo.

Para solucionar este caso, los términos de velocidad, según la tabla, se tiene que:

$$[v] = [vo] = L/T$$

La misma tabla nos da que L/T^2 para las dimensiones de la aceleración, por lo que las dimensiones de *at* son:

$$[at] = (L/T^2)\ (T) = L/T$$

En consecuencia, la expresión es dimensionalmente correcta.

Es importante tener en cuenta que el proceso de análisis dimensional solamente reemplaza el conjunto original de variables dimensionales por un conjunto equivalente, más pequeño, de variables adimensionales (los grupos). El análisis dimensional no dice cómo estas variables están relacionadas, dicha relación debe ser determinada teóricamente, por medio de la aplicación de leyes científicas básicas, o empíricamente mediante mediciones y análisis de datos. Sin embargo, el análisis dimensional es una herramienta muy poderosa, que provee una guía directa para el diseño experimental y el escalamiento, y para expresar relaciones operativas en la forma más general y útil. [6]

1.2 Las Matemáticas en las investigaciones Científicas.

En un inicio el profesional científico que trabajaba con las Matemáticas tuvo que dominar un lenguaje específico para la representación de las mismas dentro de los que se destacan TeX y Látex, utilizándole para publicar sus resultados, o tuvo que limitar su vocabulario matemático al sistema restricto de los símbolos y de las expresiones que se pueden manejar mediante algunas herramientas de tratamiento de

textos de uso general. Para hacer cómputos con un programa de álgebra por computadora tal como Mathematica, MuPAD, o Maple, necesitó dominar otra lengua; para utilizar un programa tal como MatLab o SciLab, necesitó dominar otra lengua.

Poco a poco se fueron creando herramientas para incorporar matemáticas en la computadora fácilmente y naturalmente, con la notación matemática estándar, para que todo lo entrado no fuera un arsenal de símbolos, sino verdaderos símbolos, de manera que las matemáticas puedan estar:

- Exhibidas en la pantalla.
- Pasadas a un sistema de álgebra computacional tal como MuPAD.
- Pasadas a un programa de modelación numérica tal como MatLab o SciLab o que pudieran ser utilizadas como entrada a una gran variedad de programas.

1.2.1 Codificar Información Matemática.

Inicialmente se escribía en ASCII, este formato es demasiado limitado. En 1986 apareció TEX, un lenguaje desarrollado por Donald Knuth que se volvió un método de marcado para las matemáticas, usado ampliamente hasta la actualidad. El problema con TeX es que es un sistema tipográfico, es decir, es un sistema que fija un estándar para la calidad visual de la materialización en papel del documento. Además es muy intensivo en procesamiento del rende rizado de los documentos. TeX fue la influencia más grande para MathML, y un gran esfuerzo ha sido puesto para que MathML tenga la misma calidad de representación, pero en materializaciones diversas.

1.2.2 MathML

Es un lenguaje de marcado creado para poder incluir expresiones matemáticas en la Web, superando así las limitaciones que en ese sentido posee HTML, con lo cual las expresiones pueden ser buscadas, cortadas, pegadas y reutilizadas en otros documentos o en programas de procesamiento simbólico y también representadas en múltiples modos (visuales, sonoros, táctiles).

MathML es una extensión de XML y puede ser considerada un módulo de XHTML. Tal como sucede con los editores de páginas Web, actualmente existen editores de expresiones que permiten crearlas y producir el código MathML sin que el usuario tenga que involucrarse con este. Algunos de los editores de MathML son Amaya, EzMath, Maple, Mathematica, OpenOffice, MathType y WebEQ. [2]

MathML incluye dos formas de marcado que deben producir representaciones iguales, una de ellas es llamada marcación por presentación y la otra marcación por contenido. Están relacionadas con las dos formas en que puede pensarse o leerse cualquier expresión matemática. Por ejemplo "x5" puede ser descripta en el contexto del álgebra de dos maneras: "equis exponente cinco" o "equis elevado a la quinta potencia" en el primer caso la estamos describiendo por presentación y en el segundo por contenido. (*Ver Anexo 2).*

Para satisfacer todas las diversas necesidades de la comunidad científica, MathML ha sido diseñado con los siguientes objetivos finales:

- Codificar material matemático útil para la enseñanza y la comunicación científica a todo nivel.
- Codificar tanto notación matemática como significado matemático.
- Facilitar la conversión para y desde otros formatos matemáticos, tanto el aspecto como la semántica.
- Conveniente para la interacción con software externo. Esto se refiere en particular a generadores de código o posibles intérpretes e inclusive evaluadores de expresiones.
- Ser extensible. No es posible definir toda la matemática, por lo que aquello que no quede definido y sea alguna vez necesario debe de ser posible de definirse.
- Adecuarse a plantillas y otras técnicas de edición de matemáticas.
- Ser humanamente legible, y simple para generar y procesar mediante software. [5]

1.3 Editores matemáticos.

Un editor de fórmulas matemáticas es un software usado para producir trabajos matemáticos o fórmulas. El programa tiene la función de formatear y alinear correctamente los símbolos matemáticos para componer la fórmula de una manera correcta y elegante.

Algunos editores requieren del aprendizaje de un lenguaje propio para introducir las fórmulas, mientras que otros son WYSIWYG (What You See Is What You Get: lo que ves es lo que obtienes). Los editores WYSIWYG tienen a menudo una barra de herramientas con botones con los símbolos matemáticos más usuales, en los que el usuario hace clic para añadir el símbolo al documento.

Algunos de estos editores también proporcionan la posibilidad de realizar cálculos y operaciones simbólicas, como por ejemplo Mathematica o TeX.

1.3.1 MathType.

Este producto es una versión ampliada del Editor de Ecuaciones de Word, producido por la misma empresa que este. Al instalarlo reemplaza al Editor de Ecuaciones y se agrega como nueva entrada en los menús del Word, permitiendo la edición de expresiones matemáticas y además posee una variedad de prestaciones adicionales, entre ellas tiene la opción "Export to MathPage" (Exportar a MathPage), a través de la cual genera un documento XML en el cual las expresiones están incluidas en formato MathML.

Entre otras de las características que los distinguen tenemos:

- La posibilidad de inserción de referencias a secciones y ecuaciones.
- Presenta una interfaz gráfica sencilla de usar.
- Acceso a cientos de símbolos y plantillas.
- Ecuaciones reeditables tras su inserción.
- Exportación de expresiones a código TeX, LaTeX, AMS-TeX, AMS-LaTeX y MathML.
- Posibilidad de resaltar elementos de una misma ecuación en distintos colores.

Dentro de las dificultades para utilizar este editor de ecuaciones para solucionar nuestro problema tenemos que este constituye un software privativo, no me permite realizar las modificaciones necesarias para que pueda interactuar con herramientas de análisis matemático o de Modelación Gráfica. Además de no ser multiplataforma.

1.3.2 MathML.NET Control.

Este editor de ecuaciones es un componente desarrollado para el marco de la Plataforma .NET programado con lenguaje C Sharp, posee una interfaz gráfica amigable, fácil de utilizar con posibilidades de crear cada forma imaginable de expresiones matemáticas permitiendo expórtalas a varios formatos de imagen o archivos (dpi) con la varias resoluciones y a documentos MathML. Permite dar formatos de color y tamaño en las expresiones; y brinda la posibilidad de utilizar múltiples plantillas y operadores.

Entre sus limitantes tenemos que no es multiplataforma, está limitado a Windows, además constituye un componente utilizado en la plataforma de .NET y por tanto no es posible realizarle cambios al mismo para poder integrarlo con una herramienta de análisis matemático. Además de no incluir una biblioteca de funciones que permita agilizar el proceso de edición al hacer uso de expresiones predefinidas.

1.3.3 WebEQ (WebEQ Equation Editor).

Este editor permite a los usuarios utilizar notación matemática y científica. Se basa en la tecnología de Java y de MathML y está disponible desde herramientas de Foro Debate (FD), Correos, entre otros; garantizando la creación de ecuaciones matemáticas e intercambio a través del correo o mensajes en el FD. En él las ecuaciones se identifican por un id, que lleva asignada una referencia por medio del que se pueden realizar citas a un mensaje de correo o del FD.

Este editor no permite su integración con herramientas de análisis matemáticos. No es posible tener un trabajo amplio con las ecuaciones, sólo editarlas para los usos antes expuestos. Es un editor privativo.

1.3.4 Kformula.

Este editor matemático está incluido dentro de la suite ofimática KOffice. Permite componer de forma sencilla fórmulas matemáticas que se pueden presentar en forma de imagen en la Web o en un documento, brinda la posibilidad de exportar a formatos compatibles con LaTeX, MathML entre otros. [7]

Entre las limitaciones de esta herramienta tenemos que no permite un amplio trabajo con las ecuaciones, tiene un formato muy rígido para el trabajo con las mismas, no se les puede dar un formato, no es posible destacar partes de una expresión con colores. No el uso de expresiones predefinidas. Además no es posible intégralo con una herramientas de análisis matemático.

1.3.5 OpenOffice Math.

Este editor de ecuaciones proporciona una interfaz simple para crear y editar fórmulas matemáticas. Contiene un panel de selección que contiene operadores, símbolos científicos, expresiones, etc., siendo suficiente seleccionar cualquiera de ellos mediante un clic de ratón para incorporarlo al documento. Las fórmulas creadas se pueden empotrar dentro de otros documentos. Apoya a múltiples fuentes y exporta a diferentes formatos dentro de los que se encuentra MathML.

Esta herramienta no permite su integración con herramientas de análisis matemáticos. No permite un trabajo amplio con las ecuaciones que se editan, manteniéndolas en un formato rígido.

Después del estudio y análisis de los productos descritos con anterioridad y atendiendo a los siguientes criterios: necesidad de poder reutilizar expresiones matemáticas diseñadas, salva de la información en ficheros con formato MathML para poder tener un intercambio de información con programas de tipo matemático en general, que sea multiplataforma y que a la vez sea capaz de interactuar con herramientas de modelación gráfica de sistemas biológicos se decide, hacer una herramienta computacional propia que además de brindar todas las posibilidades básicas para el trabajo con ecuaciones matemáticas, cumpla con dichos criterios creando así una herramienta amigable para el estudio de los modelos matemáticos descritos.

1.4 Tendencias y Tecnologías actuales a considerar.

A continuación se hace un breve análisis de las principales tendencias en cuanto a metodologías de desarrollo, herramientas y lenguajes de programación utilizadas para el análisis, diseño e implementación de este tipo de software.

1.4.1 Metodologías de desarrollo de Software.

Entre las metodologías de desarrollo de software más reconocidas a nivel mundial destacamos al Proceso Unificado de Desarrollo (RUP) y Extreme Programing (XP). Todo proceso de desarrollo de software es riesgoso y difícil de controlar, de ahí la necesidad de tener una metodología que nos garantice cumplir con dos cosas fundamentales: Con los Planes de producción del software y la satisfacción de nuestro cliente. El hecho de utilizar en nuestros proyectos una metodología en verdad no constituye una receta, pero no deja de ser algo de suma importancia si queremos ser competentes en el mercado del software. Como desarrolladores muchas veces a la hora de enfrentar un nuevo proyecto es necesario preguntarse ¿Qué metodología se debe usar para el desarrollo del software?, la respuesta no será evidente con "esta" o la "otra". La respuesta correcta se tendrá después de analizar la situación del proyecto, de ver lo que debemos producir teniendo respuestas al "quién", "qué", "cómo" y el "cuando" y sobre todo y más importante es saber qué alcance tendrá el proyecto.

XP es una de las metodologías más exitosas en la actualidad para proyectos cortos, consiste en una programación rápida o extrema, con la particularidad de que debe tener al usuario final como miembro del equipo de desarrollo, facilitando la comunicación entre los ente involucrados, siendo esto uno de los principales requisitos para el éxito del proyecto. XP tiende a adelantarse a los posibles errores realizando pruebas a los principales procesos. Emplea una arquitectura de software utilizando patrones o modelos estándares que garantiza la reutilización del código y desarrolla la programación en pares; a través del trabajo de dos desarrolladores en una misma estación de trabajo, mientras uno escribe el código, el otro revisa. Su objetivo principal es entregar un producto operacional inicial en un período de tres a cuatro meses.

XP no es recomendable para aplicar a este proyecto porque la misma exige la integración total del cliente al equipo de trabajo y no se tiene un conocimiento concreto de los requisitos; condiciones con las cuales no se cuenta, ya que el cliente es un agente externo al equipo de trabajo, se tienen bien definidos los requerimientos y el desarrollo del proyecto está definido para un tiempo prolongado, no relativamente corto como propone XP. Por tanto se decide adoptar el uso de la metodología de RUP.

La metodología RUP, llamada así por sus siglas en inglés Rational Unifeid Process, fue la seleccionada para el desarrollo del presente proyecto, su adaptabilidad al mismo y las condiciones de desarrollo que ofrece, garantizan el cumplimientos de las metas trazadas. Es una metodología que implementa las mejores prácticas asociadas a la Ingeniería de Software:

- Desarrollo iterativo
- Manejo de los requerimientos.
- Uso de una arquitectura basada en componentes.
- Modelización visual.
- Verificación continúa de la calidad.
- Manejo de los cambios

Brinda la posibilidad de contar con cuatro fases para el desarrollo del proyecto definiéndose en cada uno de ellas, los hitos que permiten evaluar el progreso del proyecto. Como tal se tiene la “fase de inicio” que permite tener una visión general del proyecto, la “fase de elaboración” donde identificamos la arquitectura de software apropiada al proyecto, la “fase de construcción” donde se debe alcanzar una primera parte operacional del software y la “fase de transición” donde se debe lograr el realce del proyecto. El tránsito por cada una de estas fases se realiza mediante “ciclos de iteraciones” cuyos objetivos se establecen en función de la evaluación realizada a iteraciones anteriores. Se destaca que cada ciclo de vida que se desarrolla por cada iteración es llevada bajo dos disciplinas “Disciplina de Desarrollo y Disciplina de Soporte” que garantizan el éxito del proyecto.

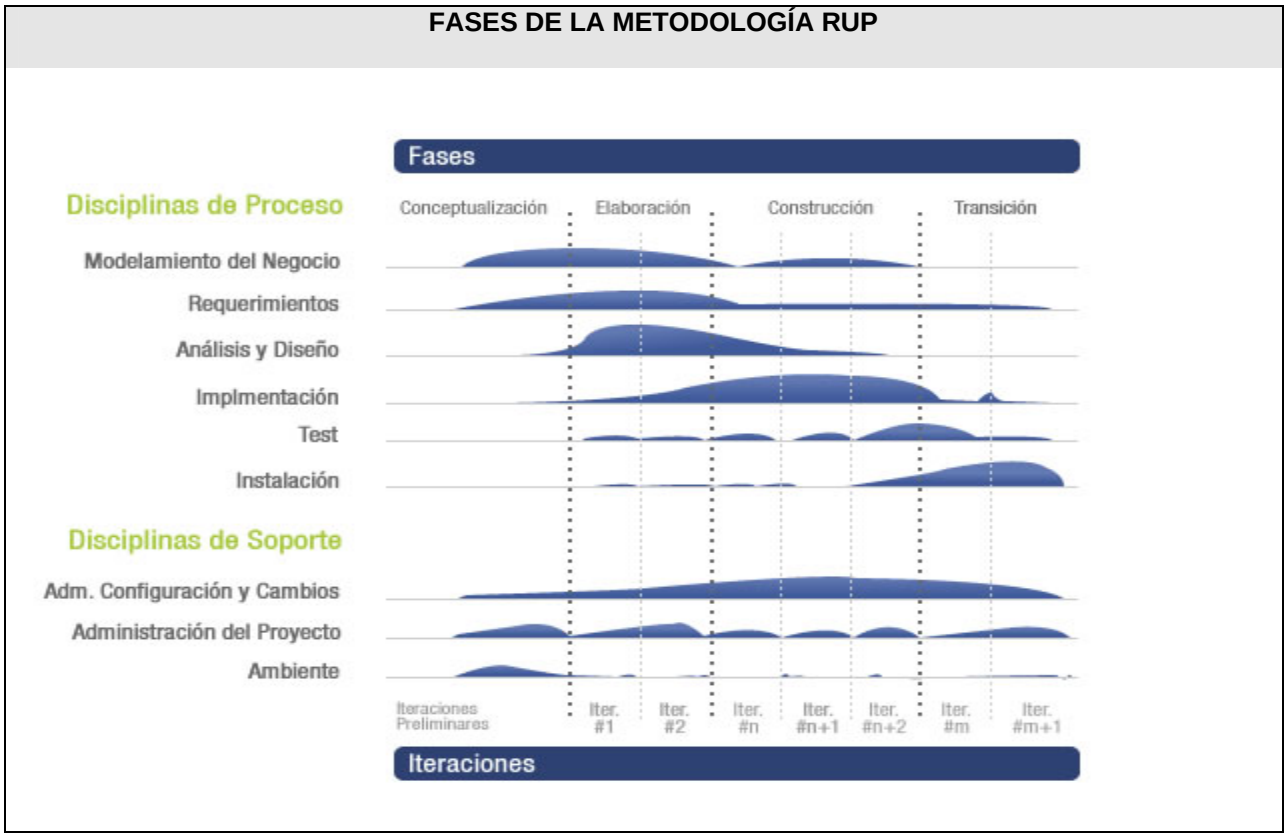


Figura 2: Fases de la Metodología RUP.

1.4.2 Lenguaje de Modelación

El Lenguaje Unificado de Modelado (UML); es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. Captura decisiones y conocimiento sobre los sistemas que se deben construir. Se usa para entender, diseñar, hojear, configurar, mantener, y controlar la información sobre tales sistemas. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. El lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar. UML incluye conceptos semánticos, notación, y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas. Está pensado para ser utilizado en herramientas interactivas de modelado visual que tengan generadores de código así como generadores de informes. La especificación de UML no define un proceso estándar pero está pensado para ser útil en un proceso de desarrollo iterativo. Pretende dar apoyo a la mayoría de los procesos de desarrollo orientados a objetos. [1]

Otras características a destacar:

- Es una de las herramientas más emocionantes del mundo actual del desarrollo de software que permite generar diseños que capturen cada una de las ideas de los desarrolladores, siendo estos de fácil compresión por las demás personas involucradas en el proceso de desarrollo del proyecto.
- Permite organizar eficientemente el proceso de diseño y construcción del software a través de diferentes modelos.
- Constituye un sistema de notación fácil que se ha convertido en un lenguaje estándar para el diseño y desarrollo de software fiable, eficiente y de calidad.
- Enfoca una tecnología orientada a objetos.
- Muestra viabilidad en la corrección de errores.
- Permite el desarrollo incremental e iterativo del software a través de los artefactos que se van creando.
- Posibilita la participación del cliente en todas las etapas del proyecto.

1.4.3 Lenguaje de Programación. Utilizado Java.

Durante las últimas dos décadas, C y C++ han sido los lenguajes más utilizados para desarrollar software. Estos lenguajes ofrecen una gran flexibilidad pero, en cambio, la productividad no es muy alta ya que requieren mucho tiempo de desarrollo. Otros lenguajes como Visual Basic de Microsoft pueden ser más productivos, pero no aprovechan todas las funciones de la plataforma que les da soporte. Con la idea de solucionar estas dificultades surge C Sharp (C#) con la perspectiva de muchos lenguajes, pero sobre todo con la de C++ y Java. [8]

Otro aspecto importante a tener en cuenta en un lenguaje de programación es la portabilidad. Poder usar la misma aplicación en distintas arquitecturas o sistemas operativos sin tener que recompilar supone un gran ahorro de desarrollo. Esta es la característica más importante de Java, esto se debe a que no es el sistema operativo quien ejecuta directamente un programa, sino la máquina virtual. Por otra parte el código de C# se ejecuta en un entorno de ejecución manejado, lo que supone el paso tecnológico más importante para hacer que C# pueda ejecutarse en distintos SO. Sin embargo algunas de las bibliotecas de .NET se basan en Windows, particularmente la biblioteca WinForms, que depende de algunos detalles fundamentales de la API.

Para el desarrollar nuestro proyecto hemos decidido optar por Java que entre otras de sus características se destacan las siguientes:

- **Es orientado a objetos**.
- **Arquitectura neutral, portátil y robusta:** Es neutral, al adoptar un sistema de código binario que es independiente de arquitecturas hardware, sistemas operativos y sistemas de ventanas. Es portátil, al definir de forma precisa los tipos y tamaños de los datos. Y es robusta, al poseer un chequeo del código tanto en tiempo de compilación como de ejecución. Y la mayor diferencia con C y C++, el modelo de memoria de Java elimina la posibilidad de sobrescribirla y la corrupción de los datos.

- **Simple:** Posee las estructuras mínimas de un lenguaje de programación tradicional, sin añadir ninguna estructura más.
- **Independiente de la plataforma:** Java se compila a un formato de código de byte que puede ser leído e interpretado por muchas plataformas, incluyendo Windows 95, Windows NT, Solaris 2.3, GNU\Linux, Mac OS, etc.
- **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.
- **Multithread (Capacidad multihilo):** En Java, todo transcurre de forma paralela, con varias tareas de forma simultánea. Un único programa Java puede procesar diferentes cosas de forma independiente y continua
- *Creación de aplicaciones distribuidas.*

1.4.4 Herramientas para el desarrollo del editor de ecuaciones.

1.4.4.1 Propuesta de desarrollo con heramientas CASE.

Las **Herramientas CASE** (**C**omputer **A**ided **S**oftware **E**ngineering, Ingeniería de Software Asistida por Computadoras) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software.

Algunas herramientas CASE conocidas son el ArgoUML, Rational Rose, Visual Paradigm, Easy CASE, Xcase, CASE Studio 2, CASEWise entre otras. Dentro de las más conocidas se encuentra el Rational Rose y el Visual Paradigm.

Rational Rose es la herramienta CASE desarrollada por los creadores de UML (Booch, Rumbaugh y Jacobson), que cubre todo el ciclo de vida de un proyecto: concepción y formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases y entregables.

Esta herramienta se destaca por sus altos precios, donde los usuario de Rose necesitan comprar muchas de las herramientas que usualmente vienen integradas con los productos de la Suite de Rational, por ejemplo, RequisitePro, SoDA, Test Manager, etc.

Visual Paradigm (VP-UML) utiliza UML como lenguaje de modelado para la construcción de los sistemas software ofreciendo soluciones de software que permiten a las organizaciones desarrollar las aplicaciones de calidad más rápido, bien y más barato. Tiene la capacidad de ejecutarse sobre diferentes sistemas operativos lo que le confiere la característica de ser multiplataforma. Integra diferentes funcionalidades para el desarrollo de aplicaciones como el modelado de UML, el modelado de base de batos, el modelado de requerimientos, el modelado del proceso de negocio, la interoperabilidad con otras herramientas CASE, la generación de documentación entre otros.

Además presenta una potente integración con el lenguaje Java, mediante su herramienta Enterprise JavaBeans (EJB) que permite el despliegue distribuido, transaccional, seguro y portable del uso. VP-UML simplifica el desarrollo de EJB generando el diagrama estereotipado de clases, el diagrama entidad-relación de la base de datos así como permite realizar reingeniería inversa de los datos.

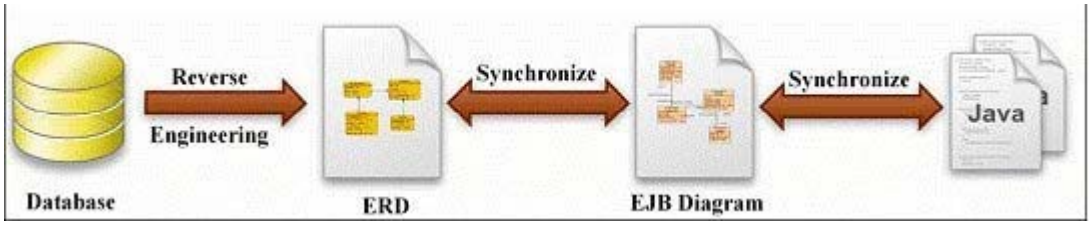


Figura 3: Visual Paradigm-UML/EJB

Se integra con las siguientes herramientas Java: Eclipse/IBM WebSphere, JBuilder, NetBeans IDE, Oracle JDeveloper, BEA Weblogic. Está disponible en varias ediciones: Enterprise, Professional, Community, Standard, Modeler y Personal.

Por el equipo de desarrollo trabajar sobre la plataforma GNU\Linux y no contar con la versión de Rational Rose que corre sobre dicha plataforma se usará para el desarrollo de la aplicación la herramienta CASE Visual Paradigm.

1.4.4.2 Entorno de desarrollo.

Los desarrolladores se han acostumbrado a las herramientas gráficas de programación, y aunque el JDK es suficiente para desarrollar Java, se han creado muchos entornos de desarrollo (IDEs) para este lenguaje entre los que podemos encontrar NetBeans, JBuilder y Eclipse.

NetBeans es una aplicación de código abierto ("open source") diseñada para el desarrollo de aplicaciones fácilmente portables entre las distintas plataformas, haciendo uso de la tecnología Java. Incluye un editor que soporta resaltado de sintaxis, herramientas para trabajar con código fuente (Java, C++, HTML, XML, etc.), plug-ins y soporte para tecnologías tales como JSP, RMI, CORBA, JINI y JDBC. Es una herramienta pensada para escribir, compilar, depurar y ejecutar programas.

JBuilder es un entorno de desarrollo para Java desarrollado por Borland, es multiplataforma, presenta gran conectividad con las bases de datos, soportando incluso CORBA. Tiene un programa de desarrollo de JavaBeans con más de 200 prediseñados.

El IDE de Eclipse emplea módulos (en inglés plug-in) para proporcionar toda su funcionalidad, a diferencia de otros entornos monolíticos donde las funcionalidades están generalmente prefijadas, las necesite el usuario o no. El mecanismo de módulos permite que el entorno de desarrollo soporte otros lenguajes además de Java así como introducir otras aplicaciones accesorias que pueden resultar útiles durante el proceso de desarrollo como: herramientas UML, editores visuales de interfaces, ayuda en línea para librerías, etc.

Se seleccionó este IDE para el desarrollo de la herramienta principalmente por su potente editor de código, la interfaz amigable que presenta para el control de versiones subversión y la existencia en el ciberespacio de una gran cantidad de plug-ins que hacen del IDE una herramienta potente. Entre otras de sus características tenemos que es una herramienta open-source y multiplataforma.

1.4.5 Herramientas para el Análisis Matemático.

1.4.5.1 Máxima.

Máxima es un completo sistema de algebra por computador escrito en Lisp con énfasis en la computación simbólica. Realiza integración simbólica, representación 3D e incluye un método para resolución ODE. Máxima esta basado en el sistema del MIT Macsyma para álgebra basada en computadores. Máxima ha sido mantenida y extendida en los últimos 15 años, pero sólo últimamente ha recibido permiso formal del DOE (Departamento de Energía de los Estados Unidos de América) de ser distribuido bajo licencia GPL como un trabajo derivado. [9]

Máxima posee las siguientes características:

- Trazado vía netmath en red.
- Cálculos en red.
- Probado para una gran cantidad de problemas.
- Depurador a nivel de código
- Fácil de extender en nuevas formas porque se tiene acceso completo al código y un acceso a Common Lisp.
- Portable a varios sistemas.
- Está bajo licencia libre GNU por lo cual permanecerá libre.
- Fue el primer sistema algebraico por computador y es uno de los mejores.

1.4.5 Conclusiones.

Al concluir el presente capítulo se ve la necesidad de desarrollar una nueva herramienta computacional para la edición de ecuaciones matemáticas lo suficientemente flexible para el investigador y que sea capaz de interactuar con otras herramientas de modelación gráfica y de análisis matemático que permita tener un trabajo interactivo y homogéneo desde un mismo entorno de trabajo. Se decide tomar el formato estándar MathML para guardar la información matemática de los sistemas biológicos en estudio. Se

selecciona RUP como metodología de desarrollo del software, al lenguaje de modelado y programación UML y Java respectivamente. Las herramientas de desarrollo seleccionadas fueron Visual Paradigm para UML y el entorno de desarrollo Eclipse. Como herramienta para desarrollar el cálculo simbólico se selecciona al Máxima.

Capítulo 2: Modelo del Dominio.

Introducción.

En este capítulo se describen los principales conceptos e entidades que existen en el Dominio. Se representan en un modelo las interacciones entre las entidades involucradas en el dominio.

2.1 ¿Por qué Modelo de Dominio?

Teniendo en cuenta que los procesos del negocio no son visibles y no es posible establecer fronteras bien definidas acorde al objeto de estudio, se desarrolla un proceso de identificación de conceptos, entidades y sus correspondientes relaciones; los cuales se agrupan en un Modelo de Objeto o Modelo de Dominio.

2.2 Definición de las entidades y los conceptos principales.

Un Modelo del dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las cosas que existen. Muchos de los objetos o clases pueden obtenerse de una especificación de requisitos. El Modelo de Dominio tiene como objetivo fundamental comprender y describir las clases más importantes dentro del contexto del sistema.

Entidades y conceptos identificados en el Dominio:

- **Investigador:** Es el papel de la persona capacitada para trabaja con la herramienta computacional.
- **Modelo matemático:** El modelo matemático que corresponde a un sistema biológico en estudio que representa de la realidad. Su objetivo es entender ampliamente el fenómeno y tal vez predecir su comportamiento en el futuro.
- **Sistema Biológico:** Representa a un conjunto de individuos, población, elementos, organismos, o compuestos químicos que interactúan entre ellos.
- **Hipótesis:** Una posible solución del problema.

- **Ecuaciones matemáticas:** Describen la variación en el tiempo de la cantidad de población y de sustancia respectivamente**.** Son originadas por cada uno de los procesos en que toma parte la población o el compuesto químico en cuestión.
- **Mathematica:** Herramienta utilizada por los investigadores para realizar los análisis pertinentes sobre el modelo matemático. Es una herramienta especializada en análisis numérico y cálculo simbólico, contiene un lenguaje de programación que nos permite crear funciones o programas simples o sofisticados.
- **Variables:** En el marco de los sistemas biológicos representan componentes de estos sistemas (poblaciones, células, químicos). Es objetivo de los investigadores monitorizar su dinámica en el tiempo.
- **Restricciones:** Estas son limitaciones impuestas a valores de las variables las cuales pueden ser de dos formas:
    - Autoimpuestas: Son asignadas por el mismo operador.
    - Impuestas: Es cuando son asignadas manualmente por el investigador.
- **Parámetros:** Son cantidades a las cuales el operador del modelo puede asignarle valores arbitrarios, los cuales se diferencian de las variables. Los parámetros una vez establecidos se convierten en constantes.

2.3 Representación del Modelo del Dominio.

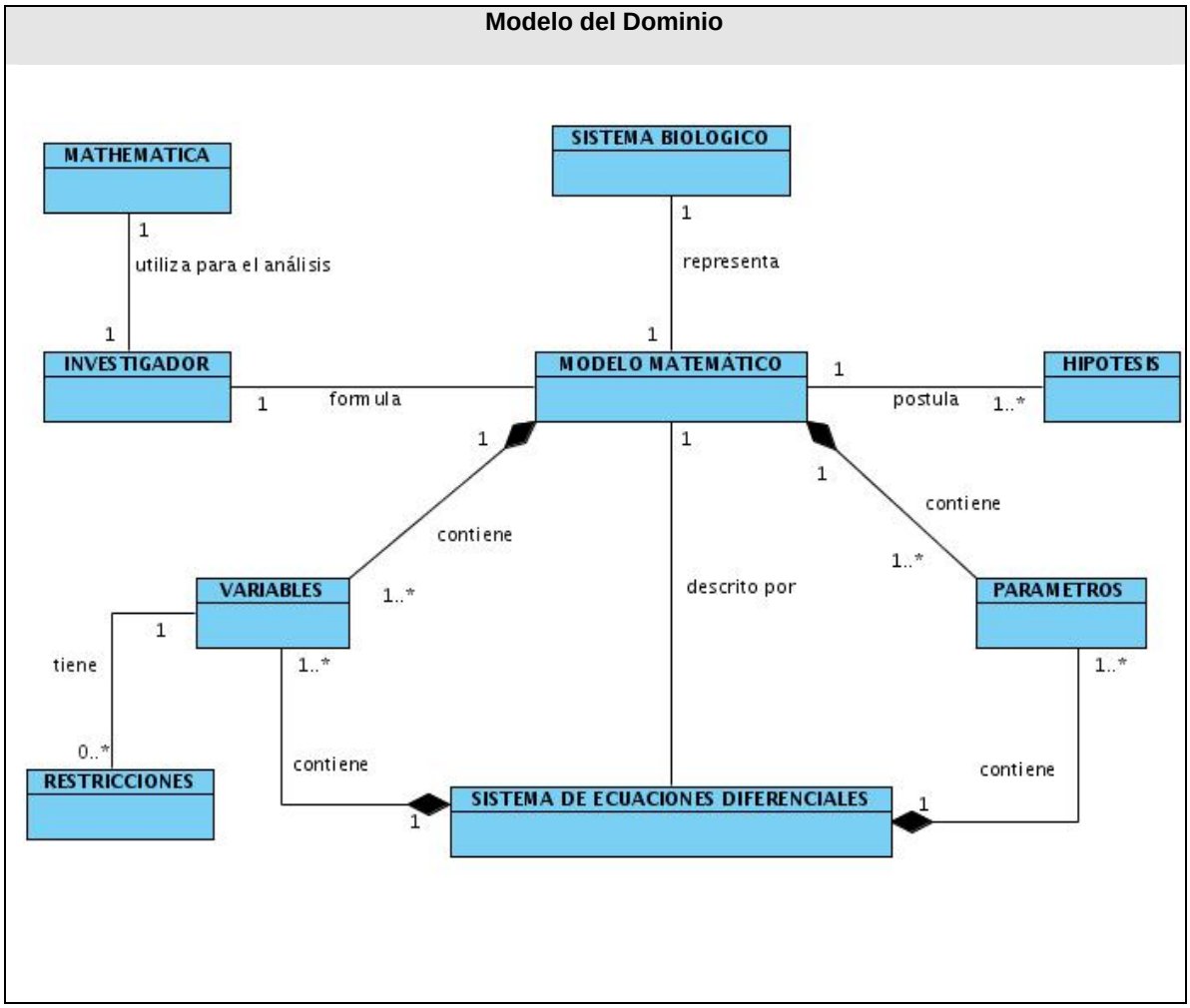


Figura 3 Modelo del Dominio.

2.4 Conclusiones.

En este capítulo se identificaron las entidades y conceptos existentes a través del Modelo de Dominio, en el que se definen 9 conceptos con sus correspondientes relaciones.

Capítulo 3: Requerimientos del sistema.

Introducción.

En el presente capítulo se describe los Requisitos Funcionales y No Funcionales del sistema, así como los Actores que intervienen en el sistema, también se encuentra el Diagrama de Casos de Uso y la descripción textual de cada uno de ellos.

3.1 Actores del sistema a automatizar.

Un Actor es un rol que un usuario juega con respecto al sistema. Es importante destacar el uso de la palabra rol, pues con esto se especifica que un Actor no necesariamente representa a una persona en particular, sino más bien la labor que realiza frente al sistema.

Nombre de autor	*Justificación*
Investigador	Es la persona capacitada para trabajar con el sistema.
Máxima	Es el sistema encargado de realizar el análisis simbólico.

Tabla 3.1: Actores del Sistema a Automatizar

3.2 Definición de los Requerimientos Funcionales.

R1. Gestionar SED.

R1.1 Crear SED.

R1.2 Modificar SED.

R2. Gestionar la biblioteca de funciones.

R2.1 Insertar nuevos elementos a la biblioteca de funciones.

R2.2 Eliminar elementos contenidos en la biblioteca de funciones.

R3. Analizar la conservación de la masa en el SED.

R4. Analizar la dimensionalidad del SED.

R5. Adimensionalizar el SED.

3.3 Definición de los Requerimientos No Funcionales.

Los requerimientos no funcionales describen alguna forma o restricción para la realización de algún requerimiento funcional o conjunto de ellos, e inclusive para todas las funcionalidades. Los mismos son los atributos del sistema, cualidades que debe tener el producto, los cuales aparecen a continuación:

3.3.1 Interfaz externa o apariencia.

La interfaz debe ser sencilla, amigable y de rápida respuesta frente a una petición del usuario de manera tal que agilice y facilite el trabajo con el software pues el sistema brindará servicios tanto a usuarios familiarizados con ambientes informáticos como a otros no familiarizados.

3.3.2 Rendimiento.

Para las nuevas funcionalidades es necesario mantener los niveles de rendimiento alcanzados por aplicaciones ya existentes, alta velocidad de procesamiento, respuesta rápida ante las solicitudes de los usuarios, además de alto grado de eficiencia y aprovechamiento máximo de los recursos en los clientes.

3.3.3 Extensibilidad.

El sistema debe ser capaz de permitir la integración con otros módulos además de permitir la inserción de cambios.

3.3.4 Mantenimiento.

Debe dar facilidad de mantenimiento, y desarrollarse los más sencilla y eficientemente posible para que en un futuro pueda ser atendido por grupos de trabajo no especializados.

3.3.5 Compatibilidad.

El sistema debe ser capaz de correr de manera independiente a la plataforma sobre el que es ejecutado. Lo cual estará asegurado por los principios del lenguaje.

3.3.6 Software.

El sistema necesitará de la instalación previa de alguna máquina virtual de Java. Una versión del software Máxima.

3.3.7 Hardware.

El requisito mínimo para ejecutar la aplicación es que el dispositivo soporte una versión de la tecnología Java™ 1.5 o superior y una capacidad de memoria de 256 Mbyte mínimo.

3.3.8 Usabilidad.

El sistema debe ser fácil de usar de manera que tenga gran aceptación entre los usuarios.

3.4 Casos de Uso del Sistema.

- Gestionar Edición SED.
- Gestionar Biblioteca.
- Analizar Adimensionalidad SED.
- Adimensionalizar SED.

- Analizar Conservación de masa.

3.5 Diagrama de Casos de Uso del Sistema.

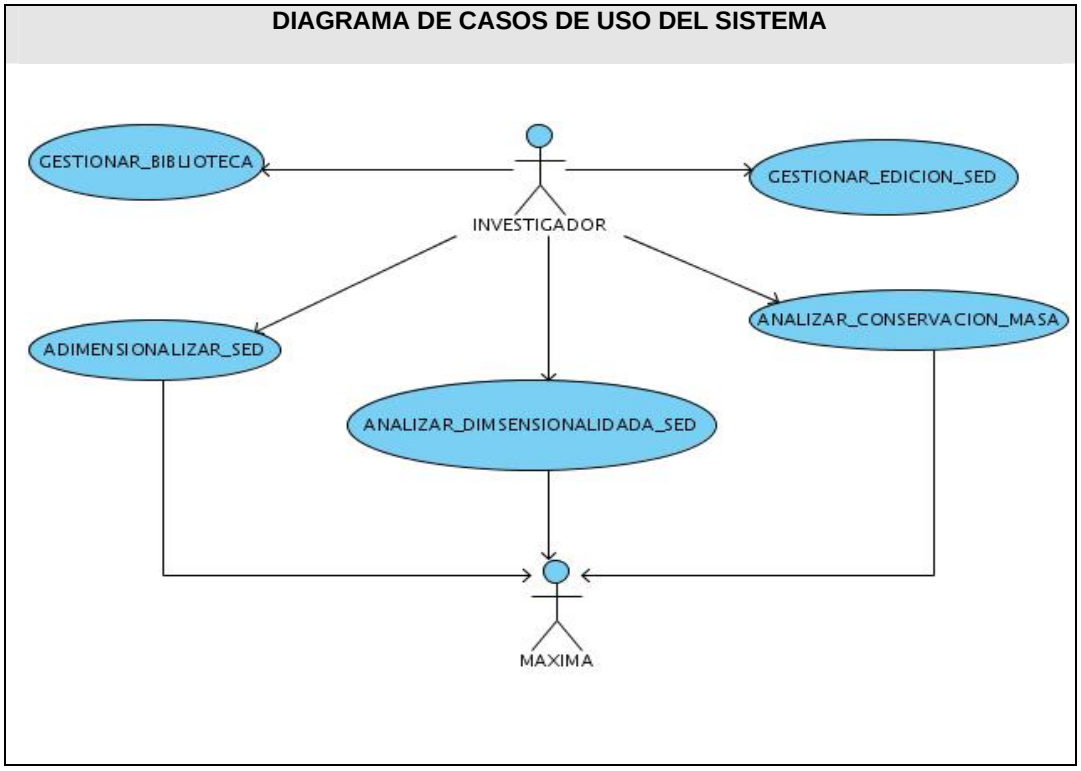


Figura 4. Diagrama de CU del sistema.

3.6 Descripción de los CU del Sistema.

Nombre del CU	Gestionar Edición SED
Actor(es)	Investigador (Inicia el CU)
Propósito	Permitir el trabajo de creación y modificación del SED en estudio.

Resumen	El caso de uso se inicia cuando el Investigador va a realizar alguna de las siguientes operaciones relacionadas con el trabajo con un SED: • Crear un SED • Modificar el SED
Referencia	R 1.1, R 1.2
Precondiciones	
Poscondiciones	Es creado o modificado un SED.
Requisitos especiales	
Curso Normal de eventos	
Acciones del Actor	**Respuesta del sistema**
1. El investigador, quiere realizar una de las siguientes operaciones: • Crear SED. • Modificar SED.	2. El sistema, en dependencia de la operación que se solicita realizar, hace lo siguiente: • Si decide crear un SED ir a Sección “Crear_SED””. • Si decide Modificar el SED ir a Sección “Modificar_SED”.

Sección "Crear_SED"	
Acciones del Actor	**Respuesta del sistema**
	1. El sistema muestra en el área de trabajo una caja vacía para conformar una ecuación correspondiente al SED.
	2. El sistema brinda la posibilidad de confeccionar dicha ecuación con elementos tales como ecuaciones predefinidas tomadas de la biblioteca de funciones, plantillas y símbolos o escribiendo variables y números a través del teclado.
3. El investigador introduce elementos a la ecuación.	4. A medida que se inserten nuevos elementos, el sistema ajustará automáticamente los tamaños de fuente, el espacio y el formato, a fin de mantener las convenciones matemáticas y tipográficas.
5. Si decide guardar Ir a evento 8.	
	6. El sistema verifica si el Sistema de ecuaciones está formado correctamente.
	7. Si el sistema de ecuaciones está correctamente formado, el sistema lo convierte en formato MathML.
	8. El sistema procede a guardar el fichero creado.

Flujo alterno de los eventos: Sección Crear_SED	
Línea 5	
5.1 Si el investigador decide introducir nuevos elementos Ir evento 3.	
Línea 5	
5.1 Si desea crear nueva ecuación marca dicha opción.	6.1 El sistema crea una nueva caja vacía para conformar una nueva ecuación.
	7.1 Ir evento 2 del flujo normal de los eventos.
Línea 7	
	7.1. Si el sistema no está correctamente creado: notifica error.
8.1 Recibe notificación de error.	
Sección Modificar_SED	
Acciones del Actor	**Respuesta del sistema**
1. El investigador marca la opción para abrir un fichero que contiene el SED.	2. El sistema muestra una interfaz para localizar dicho archivo.
3. El investigador escoge el archivo deseado.	4. El sistema verifica que archivo MathML cargado se corresponde al asimilado por el editor.

	5. Si se ajusta al formato asimilado por el editor, el sistema muestra el contenido del mismo en el área de edición.
6. Si el investigador decide insertar un nuevo elemento al sistema posiciona el cursor en la parte donde desea agregar dicho elemento.	7. El sistema brinda la posibilidad de insertar elementos tales como ecuaciones predefinidas tomadas de la biblioteca de funciones, plantillas y símbolos o escribiendo variables y números a través del teclado.
8. El investigador inserta el nuevo elemento.	9. A medida que se insertan nuevos elementos, el sistema ajustará automáticamente los tamaños de fuente, el espacio y el formato, a fin de mantener las convenciones matemáticas y tipográficas.
10. Si el investigador decide guardar los cambios Ir evento 12.	11. El sistema verifica si el Sistema de ecuaciones esté formado correctamente
	12. Si el sistema de ecuaciones está correctamente formado, el sistema lo convierte en formato MathML..
	13. El sistema procede a guardar el fichero creado.
Flujo alterno de los eventos: Sección Modificar_SED	
Línea 10	
10.1 Si decide insertar un nuevo elemento ir evento 6.	

Línea 6	
6.1 Si decide agregar una nueva ecuación al SED, solicita la creación de la misma.	7.1 El sistema crea una caja vacía representativa de una ecuación.
	8.1 Ir evento 7 del flujo normal de los eventos de la sección”Modificar_SED”.
Línea 6	
6.1.1 El investigador decide eliminar elementos del sistema de ecuaciones.	
7.1.1 Señala los elementos a elegir y marca eliminar.	8.1.1 El sistema ajustará automáticamente los tamaños de las cajas.

Tabla 3.2: Descripción CU Gestionar Edición SED.

Nombre del CU	Gestionar Biblioteca.
Actor(es)	Investigador (inicia el CU)
Propósito	Poder contar con funciones matemáticas y ecuaciones previamente creadas, además de poder guardar en ella nuevas funciones y ecuaciones para su posterior utilización.

Resumen:	El caso de uso se inicia cuando el Investigador va a realizar alguna de las siguientes operaciones relacionadas con la Manipulación de la Biblioteca de Funciones: • Eliminar la biblioteca de funciones. • Insertar nuevos elementos a la biblioteca. El sistema le muestra la interfaz correspondiente según su solicitud y ejecuta las acciones necesarias. El caso de uso finaliza cuando se obtiene un resultado de la operación efectuada.
Referencia	R 2.1, R2.2
Precondiciones	El sistema debe de haber cargado la biblioteca de funciones predefinida
Poscondiciones	Se modifica la información contenida en la biblioteca.
Requisitos especiales	
Curso Normal de eventos	
Acciones del Actor	**Respuesta del sistema**
1. El investigador, quiere realizar una de las siguientes operaciones: • Insertar nuevos elementos en la biblioteca de funciones. • Eliminar información de la biblioteca.	2.El sistema, en dependencia de la operación que se solicita realizar, hace lo siguiente: • Si decide agregar nuevos elementos a la biblioteca ir a Sección “Insertar a biblioteca”. • Si decide eliminar contenido de la biblioteca ir a Sección “Eliminar Elemento”

Sección “Insertar a Biblioteca”	
Acciones del Actor	**Respuesta del sistema**
1. Selecciona los elementos a guardar en la biblioteca.	
2. Mediante el clic derecho o la barra de herramientas selecciona la opción “Guardar como nueva ecuación”.	3. El sistema muestra una nueva ventana “Crear nuevo bloque de creación” para que el investigador defina cómo quiere que sea insertada su selección dentro de la biblioteca, defiendo los campos: nombre, categoría y descripción.
4. El investigador define tal y como serán insertados estos elementos y acepta como confirmación de la nueva inserción a la biblioteca.	5. El sistema pasa a insertar un nuevo elemento en la biblioteca de funciones.
Flujo alterno de los eventos	
Línea 4	
4.1 El investigador decide crear una nueva categoría dentro de la biblioteca para organizar las ecuaciones predefinidas por él mismo.	5.1 El sistema muestra una interfaz para que el investigador cree la nueva categoría.
6.1 El investigador introduce el nombre de la nueva categoría.	7.1 El sistema crea la nueva categoría.

8.1 El investigador define tal y como serán insertados estos elementos y acepta como confirmación de la nueva inserción a la biblioteca.	9.1 El sistema pasa a insertar un nuevo elemento en la biblioteca de funciones.
Sección "Eliminar Elemento"	
1. El usuario despliega la biblioteca para ver su contenido.	2. El sistema muestra el contenido de la biblioteca de funciones.
3. El investigador selecciona el elemento a eliminar y mediante el clic derecho selecciona la opción Eliminar.	4. El sistema muestra una ventana para que el investigador confirme que desea realizar dicha opción.
5. El investigador reafirma su deseo de realizar dicha operación.	6. El sistema elimina dicho elemento de la biblioteca de funciones.
Flujo alterno de los eventos: Sección "Eliminar Elemento"	
5. El investigador decide cancelar la acción de eliminación.	6. El sistema cancela la operación de cancelar.

Nombre del CU	Analizar Dimensionalidad SED.
Actor(es)	Investigador (inicia) y Máxima
Propósito	Comprobar la consistencia del sistema de ecuaciones comprobando si las ecuaciones que la forman son homogéneas, es decir si las dimensiones de las magnitudes a ambos lados de una igualdad son idénticas.
Resumen	El CU es inicializado por el investigador una vez creado el sistema de ecuaciones a analizar. Este sigue como pasos para el análisis dimensional el cambiar el conjunto original de parámetros de entrada dimensionales involucradas en el problema por otro conjunto de parámetros de entrada adimensionales más reducido, tomando en cuenta las dimensiones que ocupan a cada uno de estos. Luego, conjuntamente con el Sistema de ecuaciones, estos cambios son transmitidos al Máxima que se va a encargar de verificar si el sistema es o no adimensional.
Referencia	R4
Precondiciones	El sistema de ecuaciones a analizar debe de estar creado.
Poscondiciones	El sistema devuelve el resultado del análisis.
Requisitos especiales	

Curso Normal de eventos	
Acciones del Actor	**Respuesta del sistema**
1. El investigador decide analizar la dimensionalidad del sistema.	2. El sistema muestra una interfaz para que el mismo defina los cambios de los parámetros dimensionales a parámetros adimensionales, tomando aquellos definidos como básicos en el Sistema Internacional de Unidades.
3. El investigador define los cambios a parámetros adimensionales mediante combinaciones adecuadas de los parámetros dimensionales.	4. El sistema chequea que los nuevos parámetros sean realmente adimensionales.
	5. Si las transformaciones definidas son correctas el sistema pasa al Máxima el sistema de ecuaciones y los cambios definidos por el investigador.
6. Máxima realiza simbólicamente el cambio a las nuevas ecuaciones adimensionales y chequea si a ambos lados del sistema se obtienen los mismos términos.	
7. Máxima transmite al sistema el resultado del análisis dimensional.	8. Si el resultado es afirmativo, el sistema notifica al investigador.

Flujo alterno de los eventos	
Línea 5	
	5.1 Si las transformaciones definidas no son correctas el sistema notifica al investigador error en la definición de los parámetros.
6.1 El investigador recibe notificación de error.	
7.1 Si el investigador decide rehacer el proceso definiendo nuevo cambio a variables ir a evento 3 del flujo normal de los eventos.	
Línea 8	
	8.1 Si el resultado es negativo, el sistema notifica el investigador que el sistema no es consistente.
9.1 Si el investigador decide rehacer el proceso definiendo nuevo cambio a variables, ir a evento 3 del flujo normal de los eventos.	

Tabla 3.4: Descripción CU Analizar dimensionalidad SED.

Nombre del CU	Adimensionalizar SED
Actor(es)	Investigador(inicia el CU) y Máxima
Propósito	Reducir con eficiencia el número de parámetros libres en el modelo.
Resumen	El CU es inicializado por el investigador una vez comprobado que el sistema es dimensionalmente correcto. Este define un conjunto de cambios de variables de acuerdo a funciones que propicien la correcta adimensionalización del sistema con el fin de reducir el número de parámetros libres a evaluar posteriormente. El sistema y las transformaciones definidas son transmitidos al Máxima para que realice el cálculo simbólico y así obtener el sistema adimensionalizado.
Referencia	R5
Precondiciones	El sistema debe de estar creado y debe ser dimensionalmente correcto.
Poscondiciones	Se reduce el número de parámetros, obteniéndose un nuevo sistema de ecuaciones.
Requisitos especiales	
Curso Normal de eventos	
Acciones del Actor	**Respuesta del sistema**
1. Una vez editado el sistema de ecuaciones, el investigador decide reducir los parámetros mediante la adimensionalización.	2. El sistema brinda la posibilidad de efectuar esta acción.

3. El investigador define el cambio de variables para efectuar la adimensionalización.	4. El sistema chequea el cambio de variables definido sea correcto.
	5. Si las transformaciones definidas son correctas el sistema pasa al Máxima el SED y los cambios definidos por el investigador.
6. Máxima realiza simbólicamente el cambio a las nuevas ecuaciones adimensionales y las transmite al sistema	7. El sistema muestra las nuevas ecuaciones resultantes.
9. Si el investigador decide rehacer el proceso definiendo nueva adimensionalización o conjunto de nuevos parámetros agregados ir al evento 3.	
10. El investigador decide guardar la adimensionalización dada.	11. El sistema guarda la adimensionalización como información adjunta al modelo original.

Flujo alterno de los eventos	
Línea 5	
	5.1 Si el cambio de variables no es correcto el sistema notifica error.
6.1 El investigador recibe notificación de error.	
7.1 Si el investigador decide rehacer el proceso definiendo nueva adimensionalización o conjunto de nuevos parámetros agregados: Ir a evento 3 del flujo normal de los eventos.	

Tabla 3.5: Descripción CU Adimensionar SED

Nombre del CU	ANALIZAR_CONSERVACION_MASA
Actor(es)	Investigador (inicia el CU) y Máxima
Propósito	Verificar que en procesos que no involucran proliferación o muerte se conserve la masa siguiendo el Principio de Conservación de la Masa.
Resumen	Una vez creado el sistema de ecuaciones, el investigador decide comprobar si el sistema es consistente de acuerdo al Principio de Conservación de la Masa, para ello el sistema enviará el SED al Máxima para que, mediante el cálculo simbólico, pueda ser comprobado cuáles son los términos de las interacciones que no cumplen con dicho principio, para así determinar si se debe a errores en la conformación del modelo o a procesos de proliferación o muerte en los que no se cumple dicho principio.
Referencia	R3
Precondiciones	El sistema a analizar debe de estar creado.
Poscondiciones	El sistema devuelve si la consistencia es real o no.
Requisitos especiales	

Curso Normal de eventos	
Acciones del Actor	**Respuesta del sistema**
1. El investigador decide analizar si se conserva la masa dentro del SED para ello marca la opción referente a dicha acción.	2. El sistema le pasa al Máxima el SED para comprobar si se conserva la masa.
3. Máxima, mediante el cálculo simbólico, suma cada uno de los términos comunes (representativos de interacciones) que conforman las ecuaciones del SED.	
4. Máxima transmite al sistema cuáles son términos que no cumplen con dicho principio, siendo aquellos en lo que su sumatoria sea distinta de cero.	5. El sistema resalta de un color aquellos términos que cumplieron con el principio para que el investigador evalúe si se debe por: proliferación, muerte, o simplemente por error en la definición del modelo matemático.

Tabla 3.6: Descripción CU Analizar conservación de masa.

3.7 Conclusiones.

Con este capítulo se obtuvieron los requerimientos funcionales y no funcionales del sistema, fueron identificados los actores del sistema y se describieron los casos de uso del mismo, mostrando sus relaciones con los actores mediante el diagrama de casos de uso del sistema.

Describir la solución del problema y del sistema mismo antes de diseñarlo, minimiza los errores e inconsistencias, mostrando de forma clara los objetivos y funcionalidades que le conciernen. De esta forma el sistema es comprendido con mayor profundidad tanto por los clientes como por los desarrolladores.

Capítulo 4: Diseño del sistema.

Introducción.

El Diseño del Sistema permitirá conseguir una comprensión más precisa y detallada de los requisitos funcionales y no funcionales. Se describe la composición y estructura de la herramienta, que sirve como entrada apropiada y punto de partida para las actividades de implementación subsiguientes mediante la definición de los requisitos o subsistemas individuales, interfaces y clases.

4.1 Modelo del Diseño.

El modelo de diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en los requisitos funcionales y no funcionales, junto con otras restricciones realizadas en el entorno de implementación, tienen impacto en el sistema a considerar. Además, sirve de abstracción de la implementación del sistema y es, de ese modo, utilizada como una entrada fundamental de las actividades de implementación. Es representado por un sistema de diseño que denota el subsistema de nivel más alto del modelo. La utilización de otro subsistema es, entonces una forma de organización del modelo de diseño en porciones más manejables.

Los subsistemas de diseño y las clases del diseño representan abstracciones del subsistema y componentes de implementación del sistema. Estas abstracciones son directas, y representan una sencilla correspondencia entre el diseño y la implementación. [5]

4.2 Clases del Diseño.

Una clase del diseño es una abstracción sin conjeturas de una clase o abstracción similar en la implementación del sistema. Esta abstracción es sin conjeturas debido a que:

- Se especifican utilizando la sintaxis del lenguaje de programación elegido.
- Se especifica la visibilidad de los atributos y operaciones: *public, protected, private.*
- Se implementan las relaciones (referencias entre objetos) las que tienen un significado en la implementación.

- Se especifican los métodos. Tienen correspondencia directa con los métodos en la implementación.
- A menudo aparece con un estereotipo que se corresponde con una construcción de lenguaje de programación.
- Puede realizar interfaces si tiene sentido en el lenguaje de programación.

4.3 Subsistema de diseño.

Los subsistemas de diseño son una forma de organizar los artefactos del modelo de diseño en piezas más manejables. Un subsistema puede constar de clases del diseño, realizaciones de caso de uso, interfaces y otros subsistemas (recursivamente). Por otro lado, un subsistema puede proporcionar interfaces que representan la funcionalidad que exportan en términos de operaciones.

El modelo de diseño está constituido por cuatro subsistemas:

- Subsistema EDICIÓN _ GRÁFICA..
- Subsistema BIBLIOTECA _ FUNCIONES.
- Subsistema ANÁLISIS _ SED.
- Subsistema MÁXIMA..

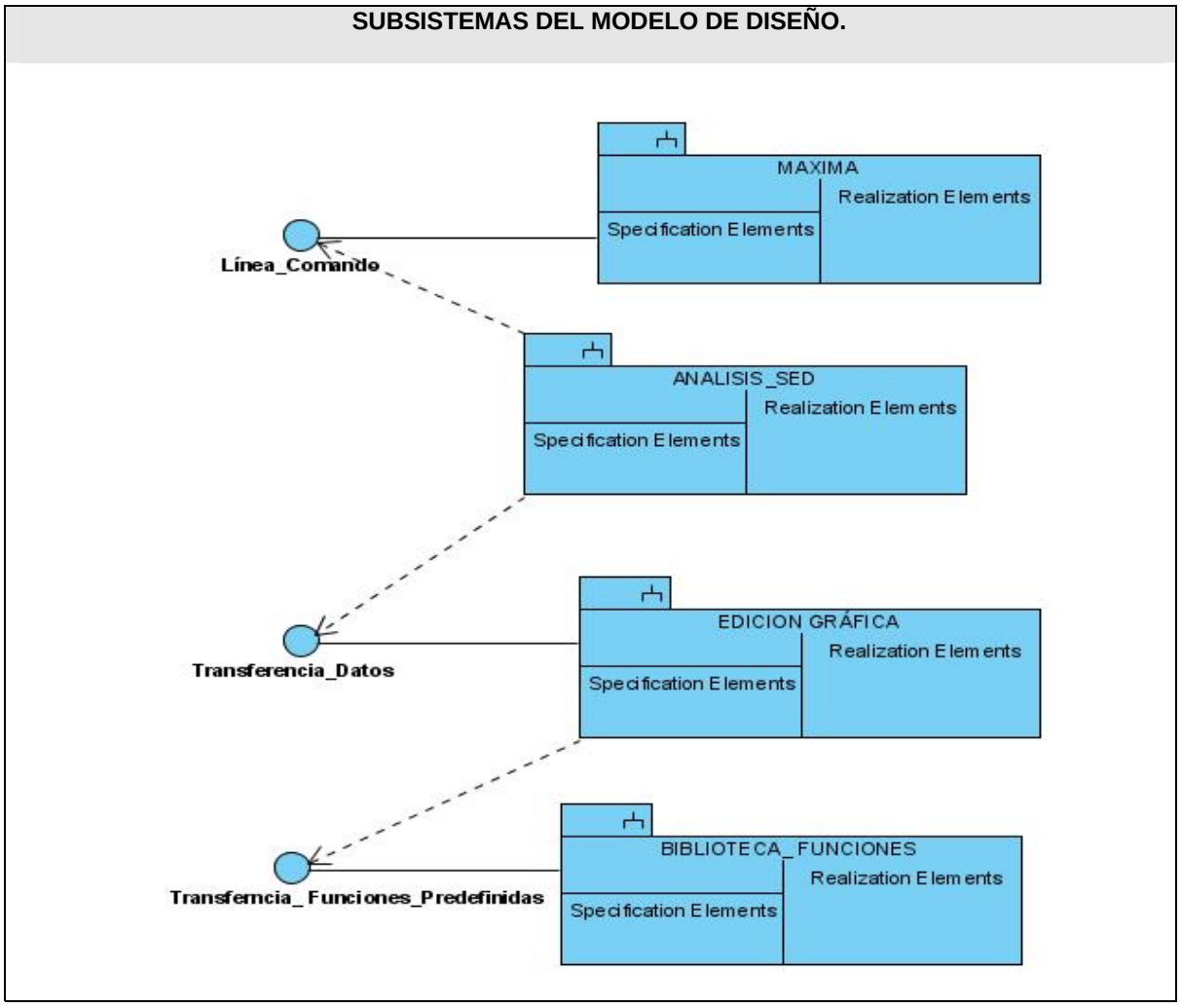


Figura 5: Modelo del Diseño.

Descripción de las clases interfaz en presentes en el Modelo de Diseño.

Nombre: transferenciaDatos	
Para cada responsabilidad:	
Nombre:	InsertarExpresAreatrabajo()
Descripción:	Transmitir a la clase controladora encargada de la edición del SED, las funciones predefinidas pertenecientes a la biblioteca de funciones, para su inserción en dicho sistema.

Tabla 4.1: Descripción clase interfaz transferenciaDatos.

Nombre: transferenciaFuncionesPredefinidas	
Para cada responsabilidad:	
Nombre:	getSistemaEcuaciones(string)
Descripción:	Pasar el sistema de ecuaciones para la realización del análisis

Tabla 4.2: Descripción clase interfaz transferenciaFuncionesPredefinidas.

Nombre: líneasComando	
Para cada responsabilidad:	
Nombre:	conectarseMáxima(string)
Descripción:	Establecer conexión con el ejecutable del Máxima.
Nombre:	adimensionalizarSED(string;list)
Descripción:	Enviar las líneas de comando para adimensionalizar el SED, enviado el mismo y la definición de los cambios de parámetros.
Nombre:	analizarConservacionMasa(string)

Descripción:	Enviar por línea de comando, el SED al que se le desea realizar dicho análisis para la obtención de su resultado.
Nombre:	analizarDimesnionalidad(string;list)
Descripción:	Enviar las líneas de comando para la realización del análisis dimensional del sistema, para la cual se necesitan el sistema y la definición de los parámetros adimensionales comprendidos en el mismo.

Tabla 4.3: Descripción clase interfaz líneasComando.

4.3.1 Subsistema Edición Gráfica.

Este subsistema es el relacionado con la edición de los modelos matemáticos de sistemas biológicos, representados por un sistema de ecuaciones diferenciales. El mismo está igualmente relacionado con la generación de un fichero MathML para el almacenamiento de dicho modelo. Las expresiones creadas serán llevadas a su forma lineal para permitir el manejo de las mismas durante los procesos de análisis a los que serán sometidas.

En el editor se genera por cada ecuación una caja (componente), la que puede contener un conjunto de cajas en dependencia de la plantilla (componente compuesto) que se vaya adicionando, además de poder contener todos los elementos básicos de las matemáticas como: símbolos y signos matemáticos y letras introducidas por el investigador.

A medida que se van añadiendo estas estructuras al editor, este ajustará automáticamente los tamaños de fuente, el espacio y el formato, a fin de mantener las convenciones matemáticas y tipográficas.

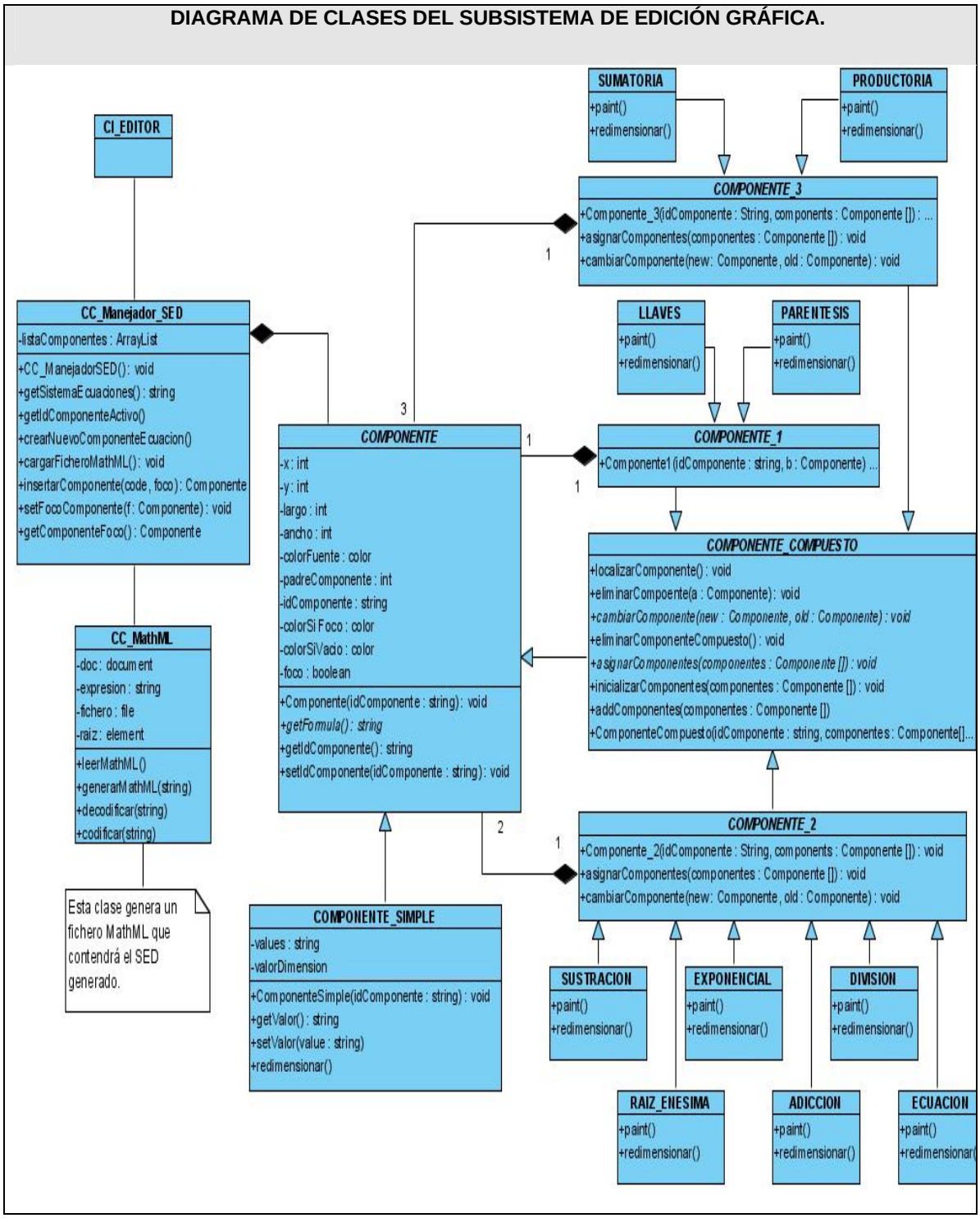


Figura 6. Diagrama de Clases del subsistema EDICIÓN _ GRÁFICA.

4.3.2 Subsistema Biblioteca de funciones.

El subsistema biblioteca de funciones corresponde a las actividades relacionadas con la manipulación de expresiones predefinidas para su posterior uso.

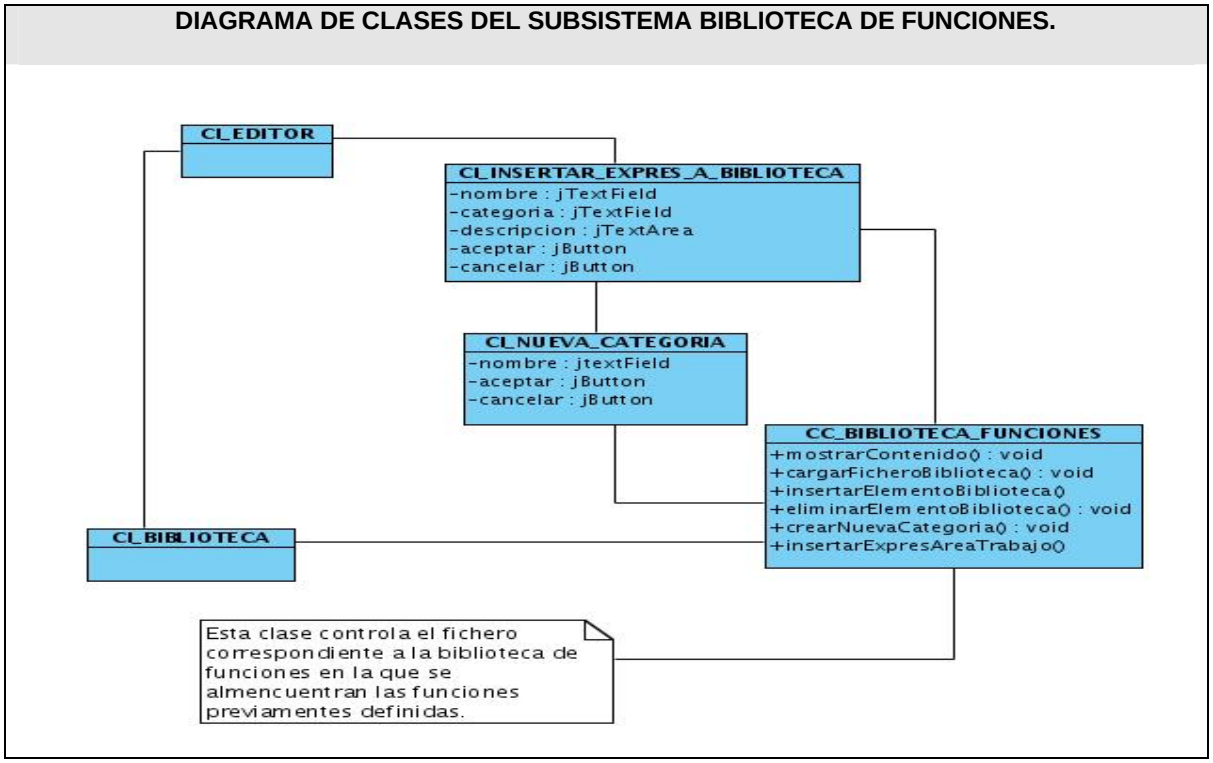


Figura 7 Diagrama de Clases del subsistema Biblioteca de Funciones.

4.3.3 Subsistema MÁXIMA.

Este subsistema representa al producto software Máxima, con el cual el sistema interactuará para la realización de los análisis pertinentes para la comprobación de la consistencia de los modelos a evaluar.

4.3.4 Subsistema Análisis SED.

Una vez generado el sistema de ecuaciones diferenciales que modela al sistema biológico el próximo paso sería comprobar la consistencia del mismo con la utilización del Máxima. Para ello se crea una instancia de la aplicación Máxima a través de la cual es posible ejecutar comandos y funciones de dicha aplicación para realizar las operaciones de análisis de dimensionalidad, conservación de la masa y adimensionalización del SED, con el paso de parámetros y del sistema generado en su forma lineal, a través de la interfaz que va a permitir realizar la transferencia de estos datos.

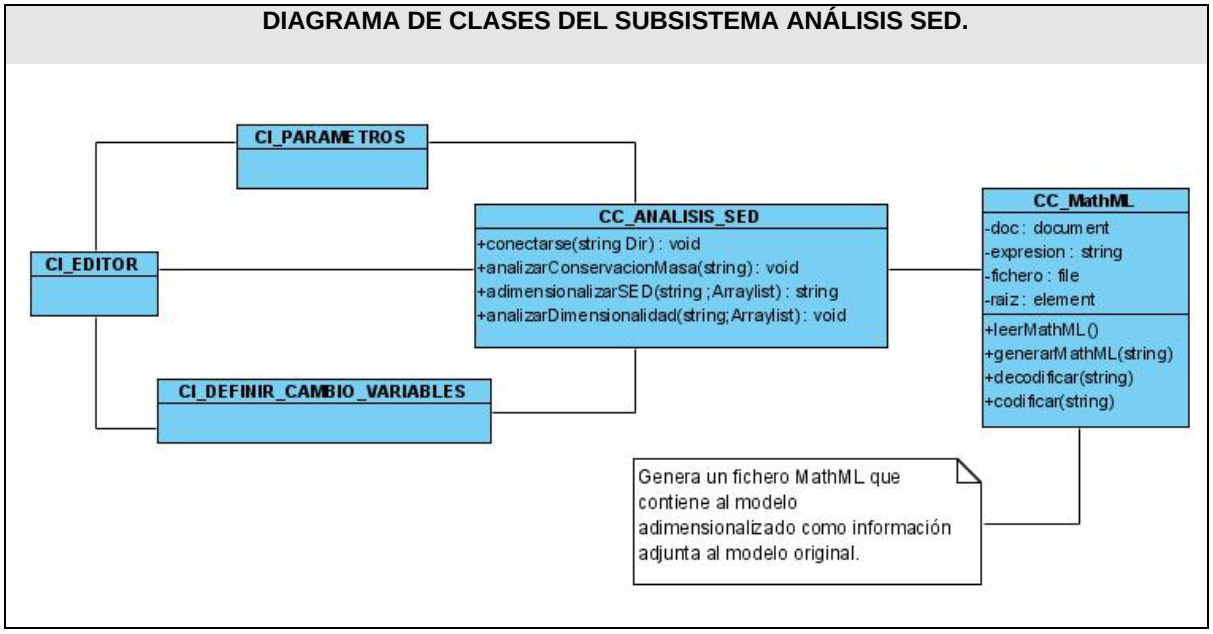


Figura 8. Diagrama de Clases del subsistema Análisis SED.

4.4 Diagramas de Interacción.

Los objetos interactúan para realizar colectivamente los servicios ofrecidos por las aplicaciones. Los diagramas de interacción muestran cómo se comunican los objetos en una interacción. Existen dos tipos de diagramas de interacción: el Diagrama de Colaboración y el Diagrama de Secuencia. El Diagrama de Secuencia es más adecuado para observar la perspectiva cronológica de las interacciones, muestra la secuencia explícita de mensajes y son mejores para especificaciones de tiempo real y para escenarios complejos.

Diagrama de Secuencia. Analizar Dimensionalidad. VER Anexo # 8

Diagrama de Secuencia. Adimencionalizar SED. VER Anexo # 9

Diagrama de Secuencia. Analizar Conservación Masa. VER Anexo # 10

Diagrama de Secuencia. Insertar Componente. VER Anexo # 11

Diagrama de Secuencia. Insertar Expresión Referenciada. VER Anexo # 12

Diagrama de Secuencia. Eliminar Componente. VER Anexo # 13

Diagrama de Secuencia. Cargar MathML.. VER Anexo # 14

Diagrama de Secuencia. Exportar MathML.. VER Anexo # 15

Diagrama de Secuencia. Insertar Elementes a Biblioteca. VER Anexo # 16

Diagrama de Secuencia. Insertar Elementos a Biblioteca con Nuevo Categoria. VER Anexo # 17

Diagrama de Secuencia. Eliminar Elemento de Biblioteca. VER Anexo # 18

4.5 Conclusiones.

Con la culminación de este capítulo queda creado el Modelo de diseño el cual servirá como esquema para la implementación del sistema a formar. Se definieron las clases pertenecientes a los subsistemas definidos en el modelo de diseño, y quedaron elaborados los diagramas de interacción del sistema.

Conclusiones

La herramienta computacional diseñada es capaz de integrar el cálculo simbólico de modelos matemáticos, conformados por sistemas de ecuaciones diferenciadas, con procesos de su edición, obteniéndose además, dentro del propio proceso de investigación, una implementación parcial del editor de ecuaciones diseñado el cual cumple con varios requisitos.

Recomendaciones.

Atendiendo a las conclusiones del trabajo se recomienda:

Implementar todas las funcionalidades diseñadas para el editor de ecuaciones e integrarla con la Plataforma de Modelación de Sistemas Biológico.

Agregar más operaciones matemáticas al editor de ecuaciones con el objetivo de ampliar su funcionalidad.

Proponer nuevos criterios de organización para el diseño la interfaz de usuario en función de lograr una mejor organización visual de los contenidos.

Referencias Bibliográficas.

1. **Schmuller, Joseph.** *Aprendiendo UML en 24 horas.* Mexico : Person Education, 2000.

2. **Plano, Ing. Jorge.** Universidad Tecnológica Nacional. *Facultad Regional Buenos Aires.* [En línea] 2 de 7 de 2004. plano@funics.org.ar .

http://209.85.165.104/search?q=cache:8S-vT4W-YDkJ:www.tecnoneet.org/docs/2004/2-72004.pdf+Por+fin:+matem%C3%A1ticas+accesibles.&hl=es&ct=clnk&cd=1&gl=cu.

3. **Osorio Ramírez, Karen.** Plataforma Computacional para el Desarrollo de la Biología de Sistemas. *Tesis de grado en opción al título de Licenciado en Ciencia de la Computación.* Universidad de la Habana : s.n., Julio de 2004.

4. **Jacobson, I., Booch, G. y Rumbaugh, J.** *El Lenguaje Unificado de Modelado.* 2000. pág. 528.

5. —. *"El Proceso Unificado de Desarrollo de Software.".* [ed.] Andrés Otero. s.l. : Imprenta FARESO, S.A., 2000. pág. 458.

6. Universidad de los Andes. *Web del Profesor.* [En línea] 24 de 2 de 2006. Recursos para Ingenieros. http://webdelprofesor.ula.ve/ingenieria/marquez/articulos/unidades.

7. The KOffice Project. [En línea] 11 de 10 de 2005. http://www.koffice.org/kformula/.

8. MSDN Español. [En línea] 13 de 1 de 2003. http://www.microsoft.com/spanish/msdn/vstudio/techinfo/articles/upgrade/Csharpintro.asp.

9. Maxima, A Computer Algebra System. [En línea] http://maxima.sourceforge.net/es/.

10. La Universitat de Cervera. *http://www.unedcervera.com.* [En línea] 28 de 12 de 2003. http://www.unedcervera.com/c3900038/estrategias/estrategias_a_dimensional.html.

Bibliografía

1. Baeza, J. J. (s.f.). *Optimización de Modelos*. Obtenido de http://www.uv.es/%7Ebaeza/optimi.html

2. Guerrero, B. A., & Founier, J. (2004). *Creacion de un espacio virtual para el intercambio de lenguaje científico.*

3. Las matemáticas una ciencia para entender la vida. (2006). Madrid, España.

4. Modelos matemáticos en medicina y biología. (1994). *Revista de Investigación Clínica* , 46(4): 307-321.

5. Polidori, D., & Trimmer, J. (2003). Acelerar la comercialización de terapias avanzadas¿Tarea de la biosimulacion? *Diabetis Voice* .

6. W3C. (19 de abril de 2007). Obtenido de http://www.w3.org/Math/Software/mathml_software_cat_editors.html

7. Jacobson, I., Booch, G. y Rumbaugh, J. *El Lenguaje Unificado de Modelado.* 2000. pág. 528.

8. *"El Proceso Unificado de Desarrollo de Software.".* [ed.] Andrés Otero. s.l. : Imprenta FARESO, S.A., 2000. pág. 458.

9. Universidad de los Andes. *Web del Profesor.* [En línea] 24 de 2 de 2006. Recursos para Ingenieros. http://webdelprofesor.ula.ve/ingenieria/marquez/articulos/unidades.

10. Design Science. [En línea] 2007. http://www.dessci.com/en/products/

11. Plano, J. (2004). *Por fin: matemcaticas accesibles.* Argentina.

12. Osorio Ramírez, Karen**.** Plataforma Computacional para el Desarrollo de la Biología de Sistemas. *Tesis de grado en opción al título de Licenciado en Ciencia de la Computación.* Universidad de la Habana : s.n., Julio de 2004.

13. Schmuller, Joseph. *Aprendiendo UML en 24 horas.* Mexico : Person Education, 2000.

Anexos

Anexo 1. Ley de Malthus:

Indica que la población humana crece a un ritmo geométrico.

Enunciado:

“Existe una tendencia universal de la población a aumentar en progresión geométrica, a menos de ser frenada por las disponibilidades de los alimentos; mientras que la producción aumenta sólo en progresión aritmética.”

Anexo 2. Ejemplo del código para generar la ecuación de ejemplo mediante notación de presentación y de contenido, según MathML.

$$(x+y)^2$$

En elementos de representación:

```
<math>
<msup>
<mrow>
<mo> ( </mo>
<mrow>
  <mi> x </mi>
<mo> + </mo>
<mi> y </mi>
</mrow>
<mo> ) </mo>
</mrow>
<mn> 2 </mn>
</msup>
</math>
```

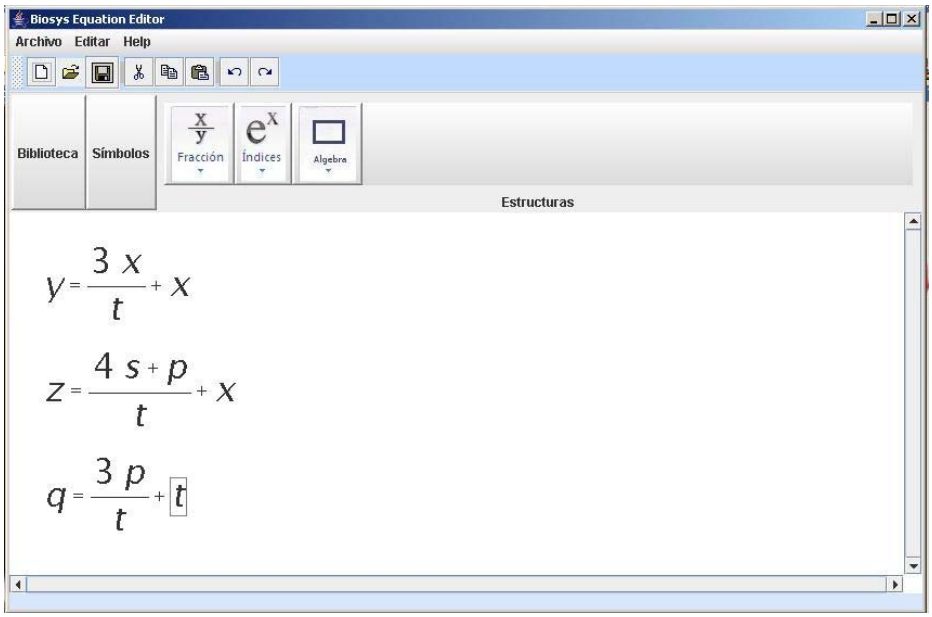
En elementos de contenido:

```
<math>
<apply>
<power/>
<apply>
  <plus/>
  <ci>x</ci>
 <ci>y</ci>
</apply>
<cn>3</cn>
</apply>
</math>
```

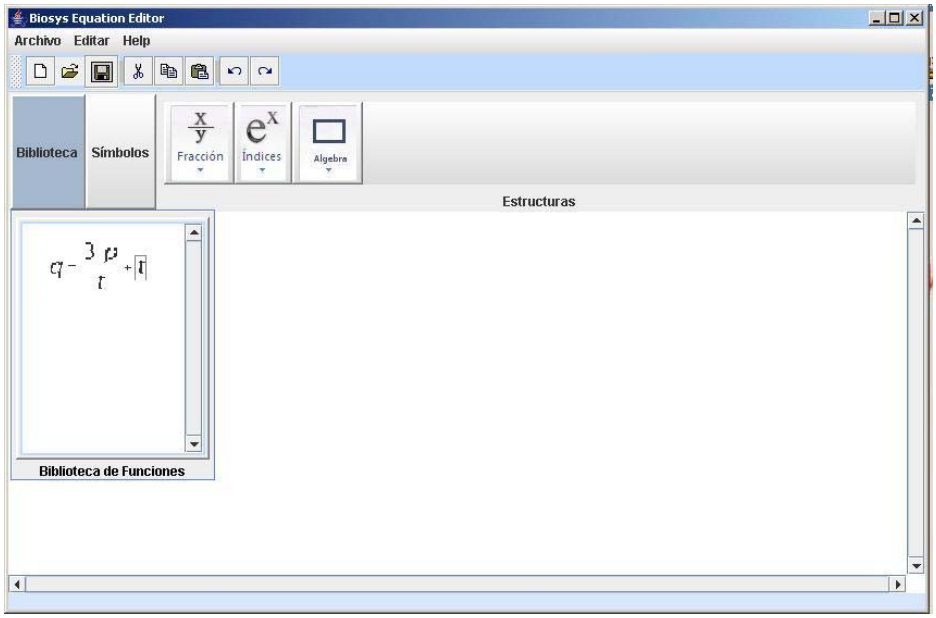
Anexo 3. Descripción de la clase interfaz CI_Editor.

Nombre: CI_Editor	
Tipo de clase : interfaz	
Atributo	**Tipo**
JtoolBarAccesosDirectos	JToolBar
JButtonNuevo	JButton
JButtonAbrir	JButton
JButtonGuardar	JButton
JButtonCortar	JButton
JButtonCopiar	JButton
JButtonPegar	JButton
JButtonDeshacer	JButton
JButtonRehacer	JButton
JPanelCentro	JPanel
JPanelHerramientasSimbolosEstructura	JPanel
JButtonBiblioteca	JButton
JTabbPaneSimbolosEstructura	JTabbPane
JScrollPaneCentro	JScrollPane
JPanelAreaPintar	JPanel

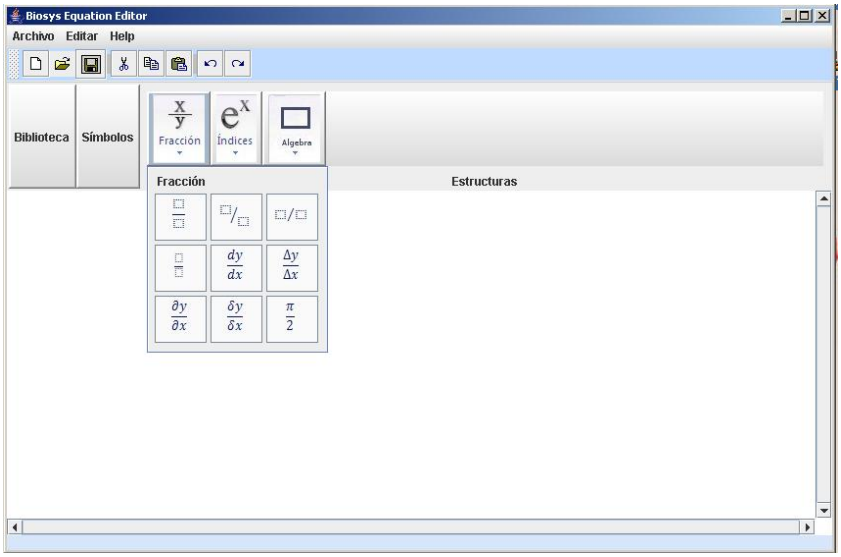
Anexo 4. Prototipo de interfaz.



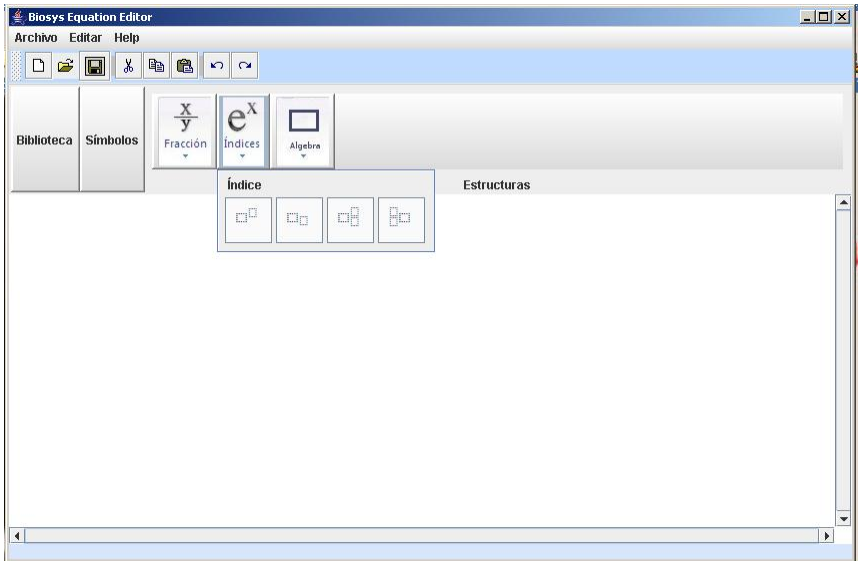
Anexo 5. Prototipo de interfaz.



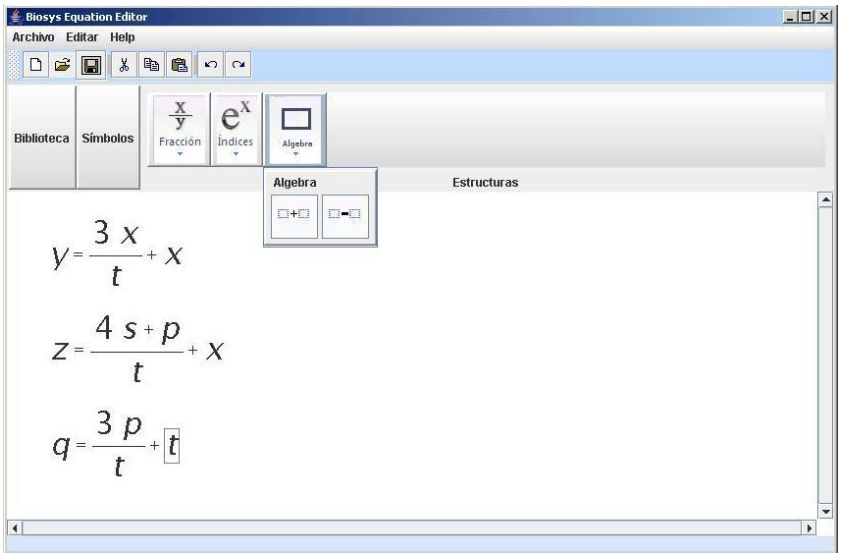
Anexo 6. Prototipo de interfaz.



Anexo 6. Prototipo de interfaz.

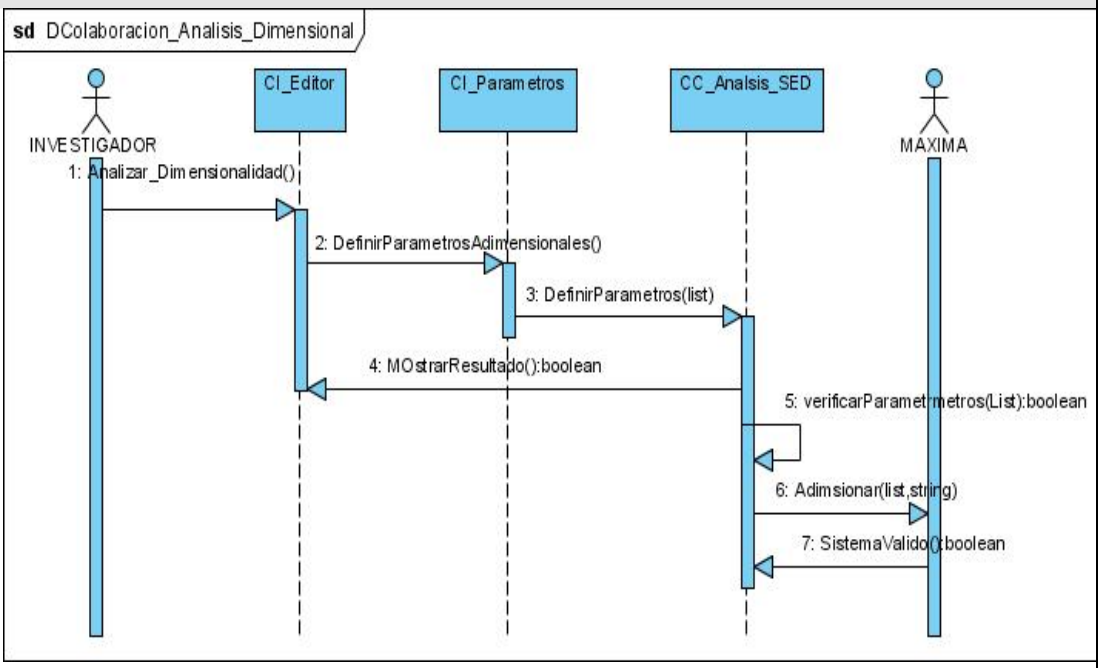


Anexo 7. Prototipo de interfaz.



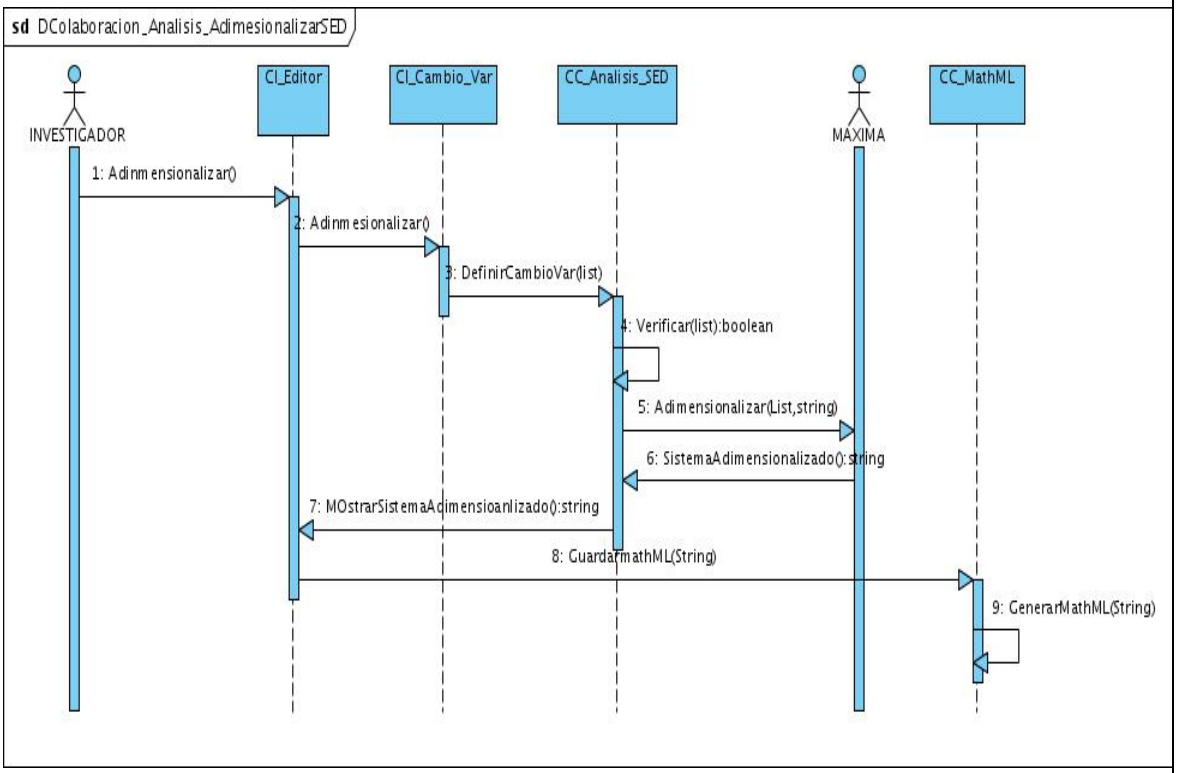
Anexo 8.

DIAGRAMA DE SECUENCIA. ANÁLISIS DIMENSIONAL.



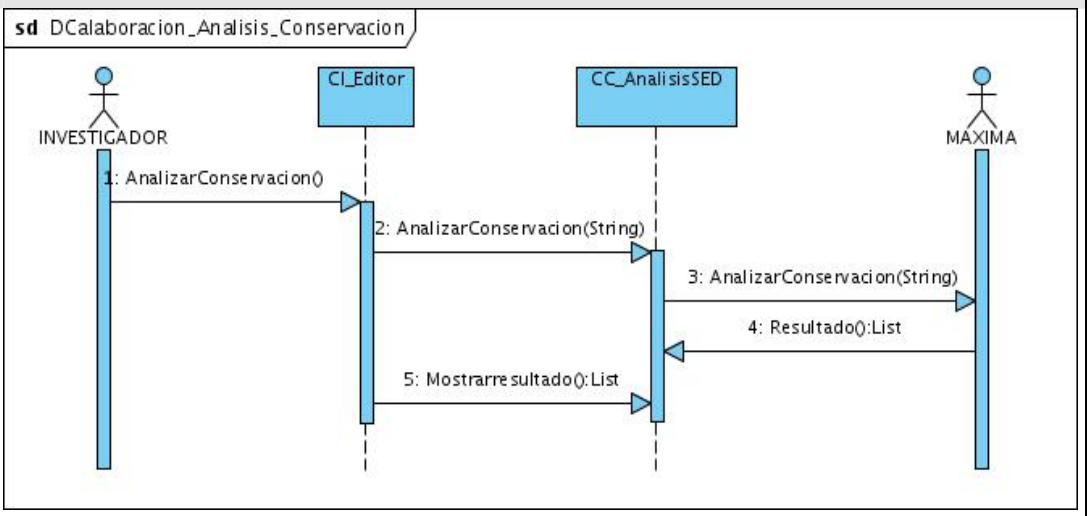
Anexo 9.

DIAGRAMA DE SECUENCIA ADIMENSIONALIZAR_SED



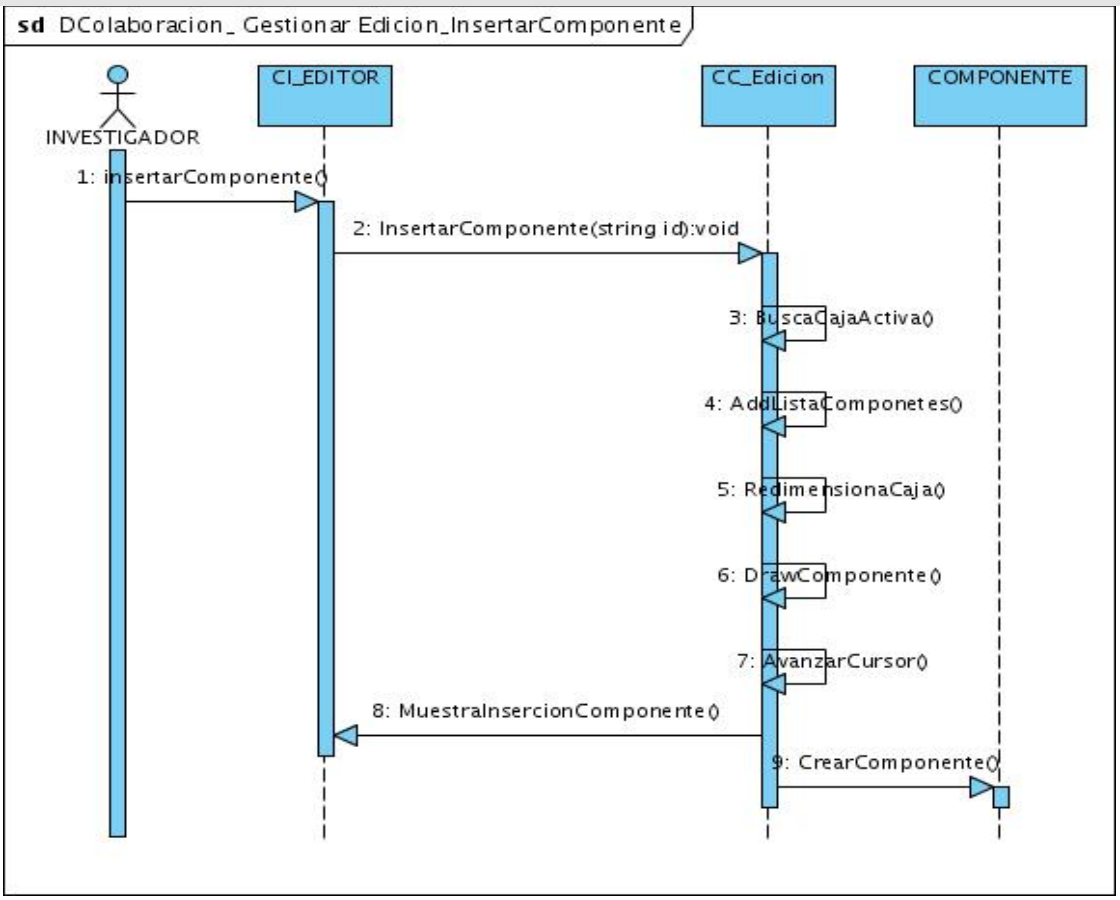
Anexo 10.

DIAGRAMA DE SECUENCIA. ANALIZAR CONSERVACION DE LA MASA.



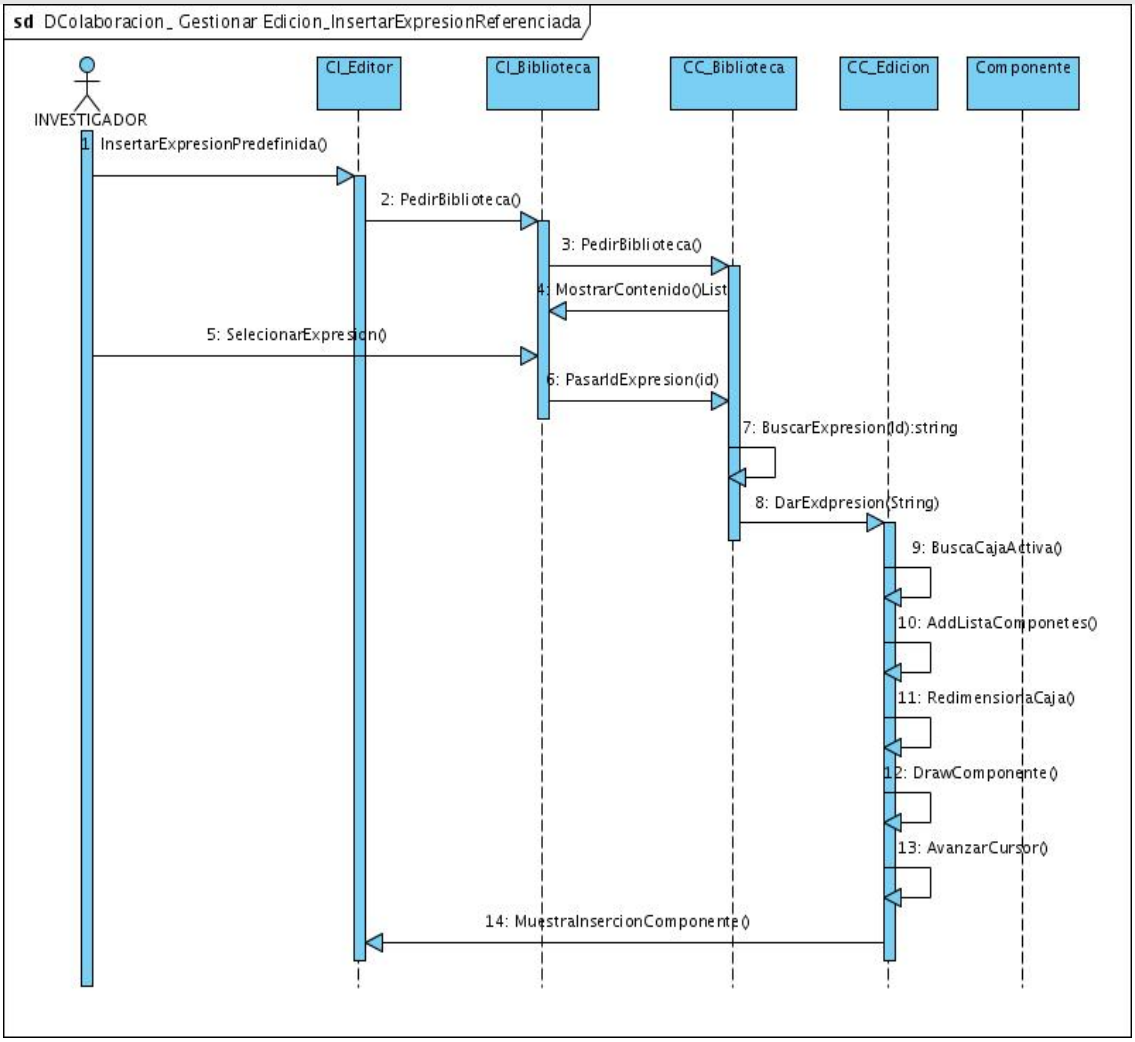
Anexo 11.

DIAGRAMA DE SECUENCIA EDICIÓN_INSERTAR_COMPONENTE



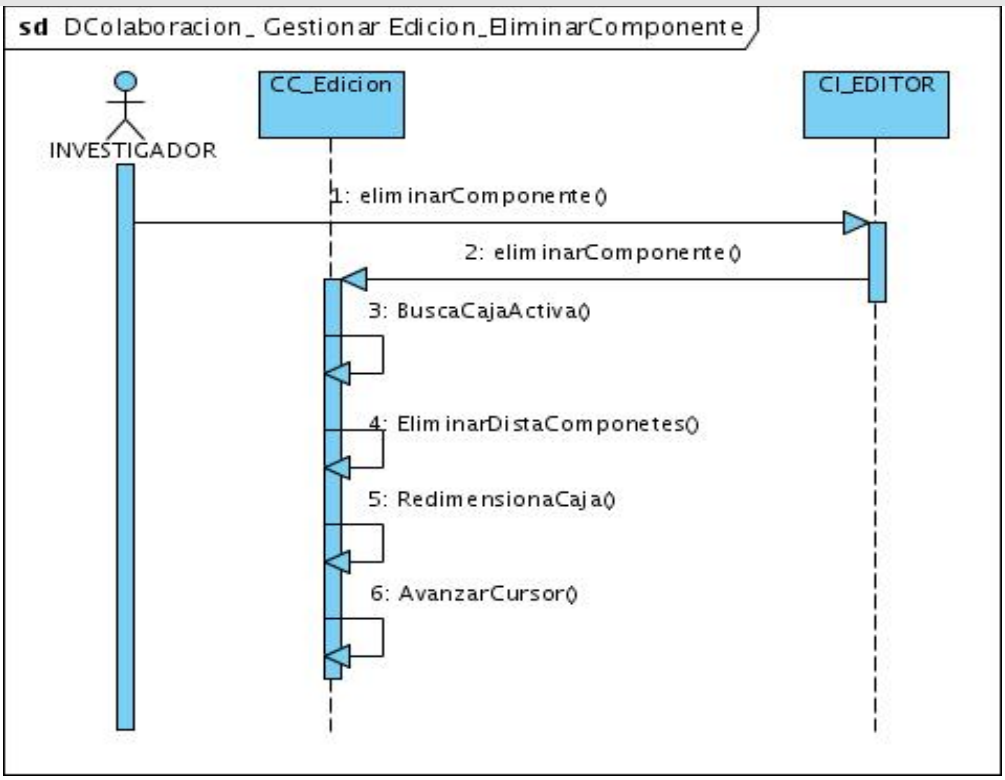
Anexo 12.

DIAGRAMA DE SECUENCIA. INSERTAR EXPRESION REFERENCIADA



Anexo 13.

DIAGRAMA DE SECUENCIA. ELIMINAR COMPONENTES



Anexo 14.

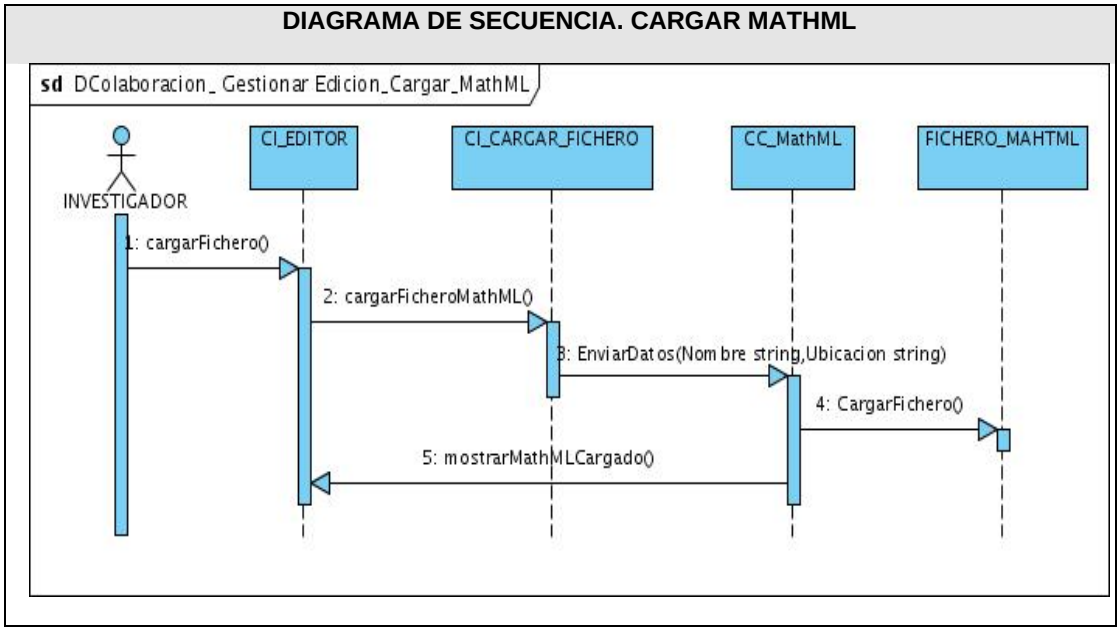
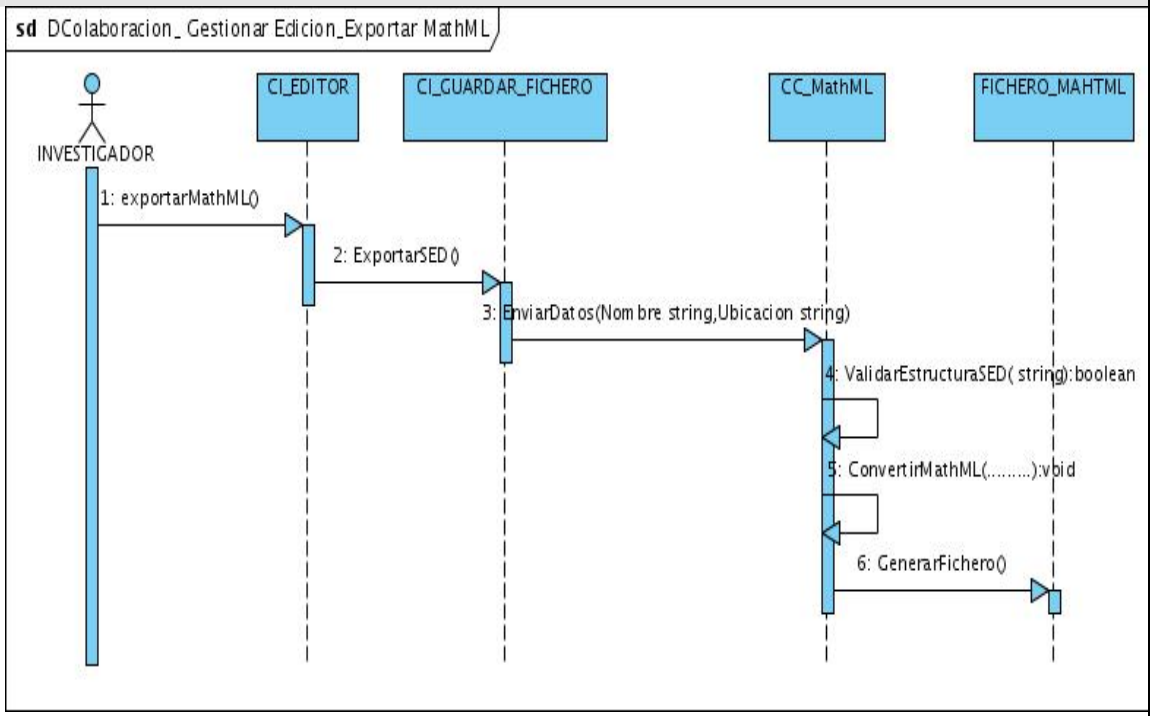


DIAGRAMA DE SECUENCIA. CARGAR MATHML
sd DColaboracion_ Gestionar Edicion_Cargar_MathML
INVESTIGADOR
CI_EDITOR
CI_CARGAR_FICHERO
CC_MathML
FICHERO_MAHTML
1: cargarFichero()
2: cargarFicheroMathML()
3: EnviarDatos(Nombre string,Ubicacion string)
4: CargarFichero()
5: mostrarMathMLCargado()

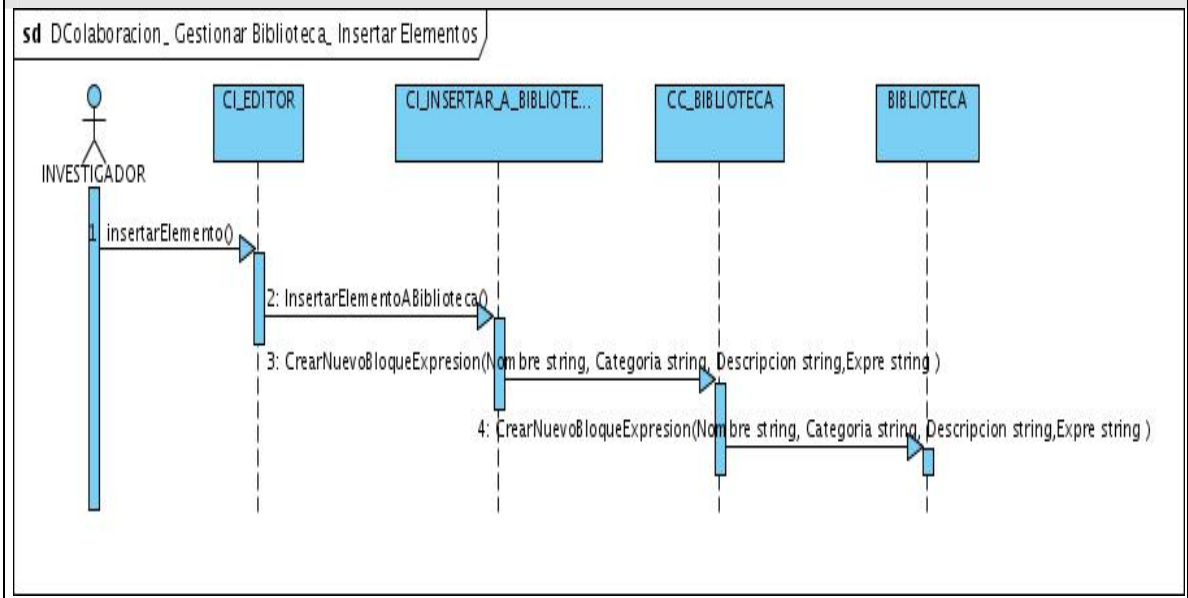
Anexo 15.

DIAGRAMA DE SECUENCIA. EXPORTAR MATHML



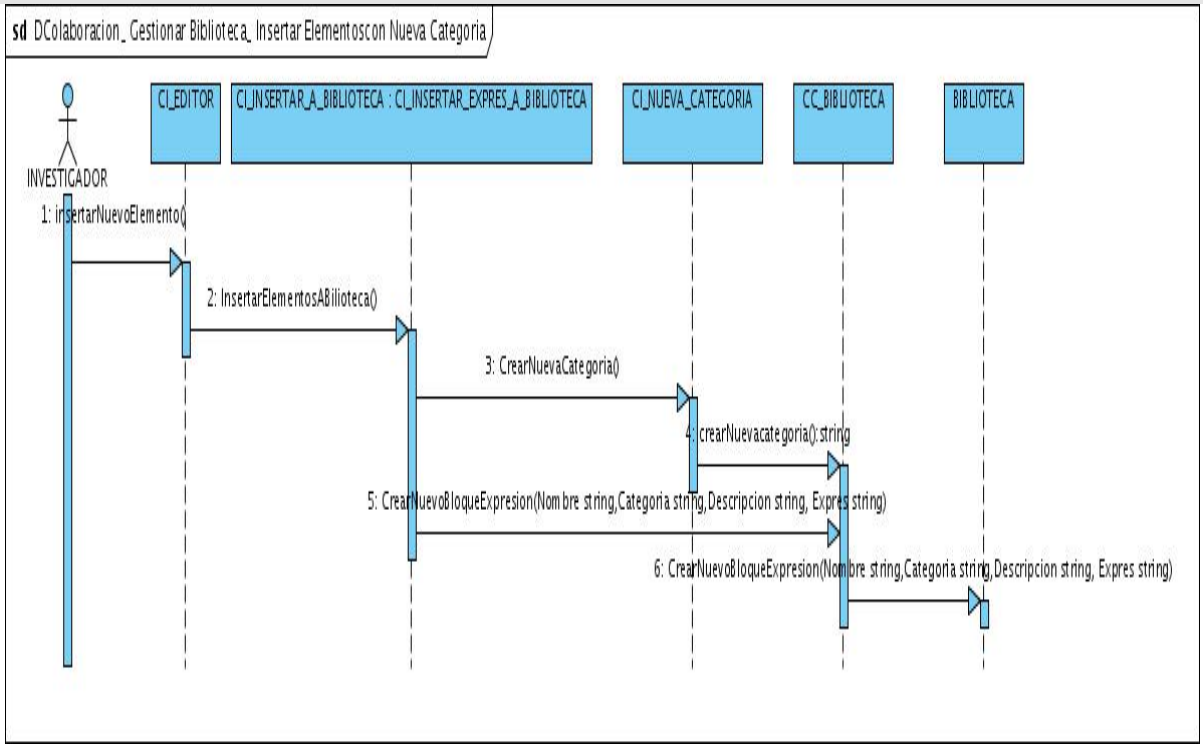
Anexo 16.

DIAGRAMA DE SECUENCIA. INSERTAR ELEMENTOS BIBLIOTECA



Anexo 17.

DIAGRAMA DE SECUENCIA. INSERTAR ELEMENTOS NUEVA CATEGORIA.



Anexo 18.

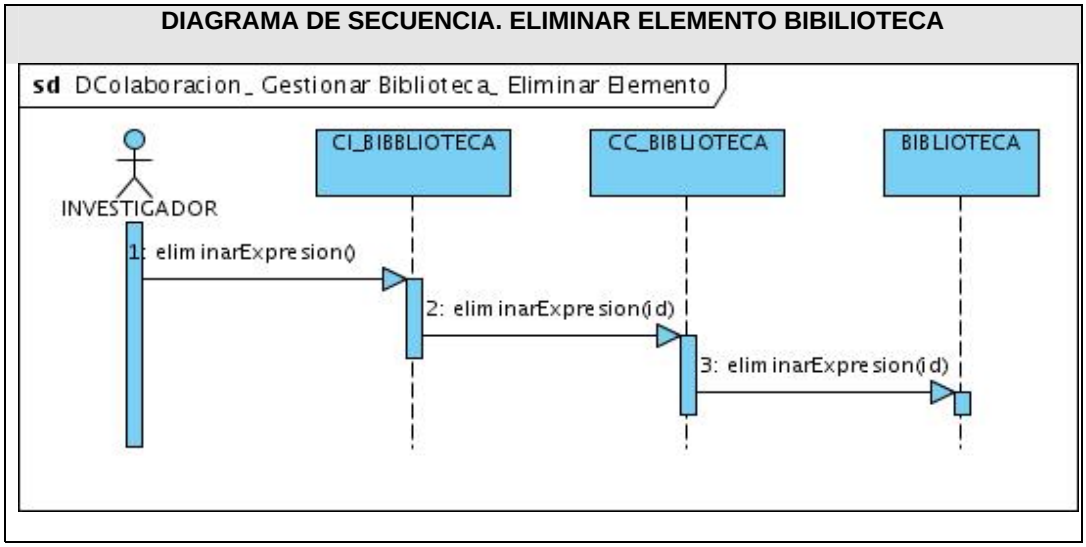


DIAGRAMA DE SECUENCIA. ELIMINAR ELEMENTO BIBILIOTECA
sd DColaboracion_ Gestionar Biblioteca_ Eliminar Elemento
INVESTIGADOR
CI_BIBBLIOTECA
CC_BIBLIOTECA
BIBLIOTECA
1: eliminarExpresion()
2: eliminarExpresion(id)
3: eliminarExpresion(id)

Glosario de Términos.

API: Representa un método para conseguir abstracción en la programación, generalmente (aunque no necesariamente) entre los niveles o capas inferiores y los superiores del software.

ASCII: Es un código de caracteres basado en el alfabeto latino tal como se usa en inglés moderno y otras lenguas occidentales.

Autómatas Celulares: Es un modelo matemático que modela a un sistema dinámico que evoluciona en pasos discretos. Es adecuado para modelar sistemas naturales que puedan ser descritos como una colección masiva de objetos simples que interactúen localmente unos con otros.

Biosimulación: Es una técnica por la que las informaciones biológicas se traducen en ecuaciones que el ordenador resuelve inmediatamente, para desarrollar medicamentos con menos coste y más eficaces en el tratamiento de enfermedades como diabetes, hipertensión o cáncer.

Cálculo simbólico: Se puede definir como la parte de la informática que diseña, analiza, implementa y aplica algoritmos algebraicos. Se caracteriza por realizar cálculos con símbolos que representan objetos matemáticos, sin errores de redondeo o truncamiento.

Catalizador: Es una sustancia (compuesto o elemento) capaz de acelerar (catalizador positivo) o retardar (catalizador negativo o inhibidor) una reacción química, permaneciendo éste mismo inalterado (no se consume durante la reacción).

Código binario: Lenguaje en el cual toda la información es representada por secuencias de ceros y unos.

Código fuente: El código fuente es un texto escrito generalmente por una persona que se utiliza como base para generar otro código con un compilador o intérprete para ser ejecutado por una computadora.

Dinámica de Poblaciones: La dinámica de una población es su desarrollo en el tiempo y en el espacio, y está determinada por factores que actúan en el organismo, en la población y en el medio ambiente. Se refiere a la dispersión, a la densidad y al crecimiento.

Farmacocinética: Es la ciencia que estudia el paso de los fármacos a través del cuerpo, cómo son absorbidos, metabolizados y eliminados.

Genómica: Rama de la biología que se encarga del estudio de los genomas. Se considera a un genoma como el conjunto de información genética (ADN de un organismo.

HTML: lenguaje de marcado diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas Web.

in silico: Significa “realizado en la computadora o vía la simulación de computadora.”

Interoperabilidad: Capacidad de un programa para acceder a múltiples sistemas diferentes.

Látex: Lenguaje para procesar textos matemáticos basado en un lenguaje de marcado formado por un gran conjunto de marcos de Tex.

Ofimática: Conjunto de técnicas y herramientas informáticas que se utiliza en la oficina. Suelen poseer herramientas de como procesadores de texto, hojas de cálculos, base de datos, presentaciones, etc.

Open-source: Término con el que se conoce al software distribuido y desarrollado libremente.

Proliferación: Multiplicación abundante de alguna cosa.//propagación, reproducción, multiplicación.

Proteómica: Puede definirse como la genómica funcional a nivel de proteínas. Es la ciencia que correlaciona las proteínas con sus genes, estudia el conjunto completo de proteínas que se pueden obtener de u n genoma.

Software privativo: Se refiere a cualquier programa informático en el que los usuarios tienen limitadas las posibilidades de usarlo, modificarlo o redistribuirlo (con o sin modificaciones), o cuyo código fuente no está disponible o el acceso a éste se encuentra restringido.

Tecnología genómica: Aplicación de la automatización de la tecnología molecular básica a un sistema paralelo masivo.

TeX: Mezcla entre procesador de textos y lenguaje de programación usado fundamentalmente para escribir documentos de contenido científico y gran calidad de impresión.

XHTML: Lenguaje de marcado pensado para sustituir a HTML como estándar para las páginas Web. XHTML es la versión XML de HTML, por lo que tiene, básicamente, las mismas funcionalidades, pero cumple las especificaciones, más estrictas, de XML

XML: Metalenguaje extensible de etiquetas desarrollado por el World Wide Consortium (W3C) que permite definir la gramática de lenguajes específicos, por lo que no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades.